Mathematical Morphology and its Applications to Signal and Image Processing

Proceedings of the 8th International Symposium on Mathematical Morphology

Rio de Janeiro, RJ, Brazil — October 10–13, 2007

Edited by

Gerald Jean Francis Banon

Divisão de Processamento de Imagem, Instituto Nacional de Pesquisas Espaciais, São José dos Campos, SP, Brazil

Junior Barrera

Instituto de Matemática e Estatística, Universidade de São Paulo, SP, Brazil

and

Ulisses de Mendonça Braga-Neto

Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX, USA

Ministério da Ciência e Tecnologia

BRASIL UM PAÍS DE TODOS GOVERNO FEDERAL

Financial Support
Conselho Nacional de Desenvolvimento Científico e Tecnológico — CNPq
Coordenação de Aperfeiçoamento de Pessoal de Nível Superior — CAPES

```
621.376.5
IN8m   International Symposium on Mathematical Morphology (8.
             : 2007 : Rio de Janeiro, RJ, Brazil).
          Mathematical morphology and its applications to signal and
       image processing : 8th International Symposium on Mathematical
       Morphology, Rio de Janeiro, RJ, Brazil, October 10-13, 2007
       / edited by Gerald J. F. Banon ...  [et al.]. - São José dos
       Campos, SP, Brazil : MCT/INPE, c2007.
          xiv + 475 p.
          Includes bibliographical references, index and author in-
       dex.
          ISBN 978-85-17-00032-4

          1. Lattice. 2. Geometry. 3. Topology. 4. Connectivity.
       5. Segmentation. I. Banon, Gerald J. F. II. Title. III. Pro-
       ceedings of the 8th International Symposium on mathematical
       Morphology.
```

This document was prepared with LaTeX and UR*Lib*Service.

Contents

Foreword

Mathematical Morphology (MM) was created in the mid-sixties by a group led by Georges Matheron and Jean Serra of the Paris School of Mines in Fontainebleau, France. By the end of the seventies, its usefulness for image analysis had been recognized in Europe, in particular within the area of microscopic imaging. Starting in the eighties, with the publication in English of Serra's books on Image Analysis and Mathematical Morphology, MM spread worldwide. In 1993, the MM community organized the first International Symposium on Mathematical Morphology (ISMM), an event that has been held approximately biennially ever since.

In Brazil, MM began to be studied at the National Institute for Space Research (INPE), in the mid-eighties, through a technical cooperation program with France. From the beginning, MM was considered a useful alternative to the linear approach to image processing. In 1987, the first master's thesis devoted to MM and its applications to Remote Sensing was presented at the INPE graduate program. Early in the nineties, MM spread over Brazil, in particular at University of São Paulo (USP), University of Campinas (Unicamp), Federal University of Minas Gerais (UFMG) and the Federal Center of Technological Education of Paraná (CEFET-PR). At that time, the first two books on MM "Bases da morfologia matemática para a análise de imagens binárias" by G. Banon and J. Barrera, and "Morfologia matemática: Teoria e exemplos" by J. Facon, were published in Brazil, in Portuguese. Since then, MM has been a very active area of research in Brazil.

In July 2005, the ISMM steering committee chose Rio de Janeiro, Brazil, among 4 four other candidates, as the site for the next ISMM edition, to be held in October 10-13, 2007. The 38 articles in this book were selected among a total of 53 submitted to the this 8th edition of ISMM. Each submitted article was reviewed by at least 2 out of 45 reviewers. The first authors of theses articles are from 11 different countries: Brazil (8), Bulgaria (1), France (15), Greece (3), Israel (1), Mexico (1), Netherlands (2), Spain (3), Switzerland (1), Sweden (2), and USA (1).

Brazil is proud to receive the MM community at ISMM'2007. The event will surely be very important in order to strengthen research in signal and image proccessing in Brazil and Latin America.

Gerald Banon, Junior Barrera, and Ulisses Braga-Neto

Preface

This edited book on Mathematical Morphology and its Applications to Signal and Image Processing is divided into eight parts containing the full papers accepted for oral presentation to the 8th International Symposium on Mathematical Morphology, held in Rio de Janeiro from October 10th to 13th, 2007.

Each part is dedicated to a specific theme of interest. The distribution of the 38 articles along these themes gives an idea of the current trends in Mathematical Morphology (MM).

In the footnote of the title page of each contribution, the reader can find the URL of the online version of that article. It may be useful to open the online version of an article when its figures contain color images, since they are all reproduced monochrome in this book.

Lattice theory

Coincidently, MM and Fuzzy Sets were both born in 1965 and during the first few decades, both communities independently made important contributions. Nowadays, we observe some convergence between them.

Bloch extends the traditional skeleton by influence zones (SKIZ) to fuzzy sets and shows a possible method of interpolation between fuzzy sets. Hirata et al. present algorithms for the computation of the basis of binary and gray-scale translation-invariant and locally-defined operators in the presence of incomplete information about the kernel. Kiselman introduces upper and lower inverses of mappings between complete lattices, analyzes their properties and links with Galois connections, and gives a thorough characterization of them, as well as an extension to upper and lower quotient of two mappings. Based on the concept of adjoint conjunctor-implicator pairs, Popov discusses fuzzy dilation-erosion adjunctions on fuzzy sets, interval arithmetic with fuzzy numbers, and a fuzzy hit-or-miss transform. From a partial-order relation defined on tree representations of images, which provide a complete inf-semilattice structure, Vichik et al. derive self-dual morphological operators that can be used as filters for noise reduction.

Geometry and topology

Despite the fact that discrete geometry is a separate field of research, it has a large intersection with MM whenever one considers the applications of MM to digital image processing.

After recalling the definitions and properties of smallest-neighborhood spaces and the Khalimsky topology on $\mathbb{Z}^n$, Melin introduces a binary operation on topological spaces called join, and shows that under local finiteness theses spaces can be uniquely decomposed as a join of indecomposable spaces. Serra investigates the conditions for generating granulometries on $\mathbb{Z}^n$ using Steiner sets, as well as conditions on these sets for them to be convex or connected. In a second article, Serra proposes a new random closed set model, combining the generation of random germs and random sets, and applies this model to the analysis of data on fires that occurred in Malaysia.

Signal processing

MM based on lattice theory is a powerful approach to solve nonlinear problems in signal processing. At the heart of this approach are the nonlinear operators of dilation and erosion as illustrated in a few of the contributions below.

Dorini and Leite introduce a scale-space toggle operator, prove some of its properties, and illustrate its usefulness for image segmentation. Karantzalos compares the construction of nonlinear scale space image representations derived from multiscale levelings constrained by four different markers. Maragos and Evangelopoulos introduce alternative approaches based on levelings and texture energy for decomposing an image into cartoon and texture components.

Image processing

The main application of MM is Image Processing. The success of MM in this field comes from the fact that, diferently from the linear approach, it is able to manage the finiteness of the digital image range. The contributions below include content-based image indexing and retrieval, texture classification, image simplification, crest line regularization, and image interpolation for computer-aided animations.

Andaló et al. introduce a new shape descriptor based on tensor scale, show how it can improve content-based image retrieval and provide a faster computation of tensor scale by exploiting the Euclidean Image-Foresting Transform. Aptoula and Lefèvre combine the complementary information extracted by morphological covariance and granulometry in order to improve texture classification, and propose an extension to color data introducing an ordering based on luminance and saturation. Meyer and Angulo introduce the so-called bilevelings of an image defined on an hexagonal grid, prove that they can be characterized in term of so-called micro-viscous operators that appear to form adjunctions between set of functions defined on vertices and edges, and verify that they lead to higher levels of simplification than with ordinary levelings. Retornaz and Marcotegui present a robust technique using morphological numerical residues for automatic localization of textual information in general image database through connected component extraction. Vachier and Meyer enlarge the concept of viscous transform to increasing operator, introduce three viscous dilations and show how dotted thin crest lines in gray-tone images may be reconnected and smoothed using these transformations. Vidal et al. extend to mosaic images their previous work on an interpolation technique for binary images based on morphological median sets, by processing all the levels of the hierarchical region-based tree that represents the region structure of the mosaic image.

Connectivity

Among the major developments in MM in the last two decades are the interrelated subjects of connectivity classes and connected operators. Connectivity classes, in their general lattice-theoretical formulation, not only unify in a single axiomatic framework useful but disparate notions of connectivity, but also include new interesting definitions of connectivity not previously possible. Connected operators, on the other hand, have become very popular in image analysis applications due to the fact that these operators can preserve edge information by working at the level of the image flat zones, which are defined using connectivity criteria.

Crespo examines in detail the equivalence between two important classes of connected operators, namely, set levelings and adjacency-stable operators. Dimiccoli and Salembier propose the use of image inpainting to improve the appearance of images processed by connected operators. Naegel et al. propose a general definition of vector-attribute filters for gray-level images and describe its application in detection and segmentation tasks.

Watershed segmentation

The watershed method represents a powerful paradigm in image segmentation. Ever since its introduction three decades ago, it has attracted the interest of many investigators and practitioners, and research activity in this area continues unabated today.

Alléne et al. study the links that exist between the watershed and minimum cuts, minimum spanning forests, and shortest-path forests. Angulo and Jeulin propose a watershed-based stochastic segmentation methodology. Audigier and Lotufo use the Image Foresting Transform and the Tie-Zone concept to establish the relationship between several discrete watershed transforms. Ten Caat et al. address the visualization of electroencephalography (EEG) data by means of a watershed-based clustering method. Cousty et al. study several properties of watersheds in edge-waited graphs. Papa et al. present a classification method based on optimum-path forests. Stawiaski et al. present a technique to compute approximate geodesics and minimal surfaces.

Texture and geometrical segmentation

The articles in this part further examine the segmentation problem, by using geometric and texture criteria, within or without a watershed framework.

Angulo presents a method to calculate texture gradients and use them for joint color and texture segmentation. Consularo et al. present an efficient method for interactive image segmentation based on inexact graph matching. Cord et al. propose morphological and linear approaches to the characterization and segmentation of random textures. Noyel et al. introduce new connectivity classes that are able to improve flat-zone segmentation of hyperspectral images. Sofou and Maragos investigate problem of image segmentation via PDEs, focusing on a generalized flooding procedure using geometric and textural information.

Algorithms and architectures

The last, but not by any means least, part of the book is devoted to an aspect of MM that allowed it to become widespread and popular in both research and practice, namely, efficient algorithms and architectures for implementation of MM operators.

Bergo and Falcão introduce an algorithm that expresses any Imaging Foresting Transform operator as a series of independent computations, facilitating thus parallel implementations of such operators. Menotti et al. present a linear time and space algorithm to compute the component tree of one-dimensional signals, from which they derive an efficient gray-level image multithresholding method. Ouzounis and Wilkinson present a concurrent implementation of a previously developed Max-Tree algorithm, which implements anti-extensive attribute filters based on second-generation connectivity. Vaz et al. describe an efficient method to implement MM operators using convex and symmetric structuring elements, including Euclidean discs and spheres.

Gerald Banon, Junior Barrera, and Ulisses Braga-Neto

Acknowledgments

The Scientific Editorial Committee would like to thank the 45 reviewers who dedicated their time to read their assigned full papers and fill out the reviewer sheets. Based on their reviews the Scientific Editorial Committee was able to rank the submissions and select the best contributions. Furthermore, their comments and recommendations were valuable to the authors in improving the initial version of their full papers.

List of reviewers

Arnaldo Araújo	Gilles Bertrand
Isabelle Bloch	Gunilla Borgefors
Michael Breuss	Michael Buckley
Bernhard Burgeth	Fábio Cappabianco
Roberto Cesar Jr.	José Crespo
Etienne Decencière	Marcos d'Ornellas
Alexandre Falcão	Franklin Flores
Irena Galic	John Goutsias
Ana Graciano	Ferzolli Guimarães
Marcelo Hashimoto	Ronaldo Hashimoto
Nina Hirata	Roberto Hirata Jr.
Dominique Jeulin	Renato Keshet
Christer Kiselman	Sébastien Lefèvre
Neucimar Leite	Roberto Lotufo
Petros Maragos	David Menotti
Paulo Miranda	Laurent Najman
Nicolas Passat	Pedro Pina
Jean-F. Rivest	Jos Roerdink
Christian Ronse	Guilherme Ruppert
Philippe Salembier	João Soares
Pierre Soille	Hugues Talbot
Iván Terol Villalobos	Marc Van Droogenbroeck

Joachim Weickert

Gerald Banon, Junior Barrera, and Ulisses Braga-Neto

I

LATTICE THEORY

An extension of skeleton by influence zones and morphological interpolation to fuzzy sets

ISABELLE BLOCH

Ecole Nationale Supérieure des Télécommunications (GET - Télécom Paris), CNRS UMR 5141 LTCI, Paris, France
`Isabelle.Bloch@enst.fr`

Abstract While the notions of influence zones and skeleton by influence zones (SKIZ) have many important applications, they have not been extended to fuzzy sets until now. The aim of this paper is to fill this gap and to show the potential usefulness of such an extension. The proposed definitions are based on fuzzy dilations and their interpretations in terms of distances. As another contribution, we show how this notion can be used to define a fuzzy median set, and a series of fuzzy sets interpolating between two fuzzy sets.

Keywords: fuzzy sets, fuzzy skeleton by influence zones, fuzzy median set, interpolation between fuzzy sets.

1. Introduction

Despite the interest of notions of influence zones and skeleton by influence zones (SKIZ), surprisingly enough they have not really been exploited in a fuzzy context until now. If knowledge or information is modeled using fuzzy sets, it is natural to see the influence zones of these sets as fuzzy sets too. The extension of these notions to the fuzzy case is therefore important, for applications such as partioning the space where fuzzy sets are defined, implementing the notion of separation, reasoning on fuzzy sets (fusion, interpolation, negotiations, spatial reasoning on fuzzy regions of space, etc.), motivating the work presented in this paper.

The first contribution is to propose definitions of notions of influence zones and skeleton by influence zones for fuzzy sets. Both influence zones and the SKIZ are then fuzzy sets, defined on the same space. The proposed definitions rely on formal expressions of the SKIZ in terms of distances and morphological dilations. Let $\mathcal{S}$ be the underlying space, endowed with a distance d, and X be a subset of $\mathcal{S}$ composed of several connected components: $X = \bigcup_i X_i$, with $X_i \cap X_j = \emptyset$ for $i \neq j$. The influence zone of X_i, denoted as $IZ(X_i)$, is defined as [15, 18]:

$$IZ(X_i) = \{x \in \mathcal{S} \; / \; d(x, X_i) < d(x, X \setminus X_i)\}. \tag{1}$$

The SKIZ of X, denoted as SKIZ(X), is then given by:

$$\text{SKIZ}(X) = (\bigcup_i IZ(X_i))^c.$$

Let us denote by δ_λ the dilation by a ball of radius λ, and ε_λ the erosion by a ball of radius λ. Then the influence zones can be expressed as:

$$IZ(X_i) = \bigcup_\lambda (\delta_\lambda(X_i) \cap \varepsilon_\lambda((\cup_{j \neq i} X_j)^c)) = \bigcup_\lambda (\delta_\lambda(X_i) \setminus \delta_\lambda(\cup_{j \neq i} X_j)) \,. \quad (2)$$

These two expressions of influence zones, in terms of morphological dilations on the one hand and in terms of distances on the other hand, constitute the basis for the proposed definitions in the fuzzy case (Section 2).

The second contribution (Section 3) is to exploit the notion of fuzzy SKIZ to define the median fuzzy set of two intersecting fuzzy sets. The iterative application of the median set computation leads to the construction of a series of interpolating sets from one fuzzy set to another one. To our knowledge, this idea of interpolation between fuzzy sets is also novel.

2. Fuzzy influence zones and fuzzy SKIZ

While several notions involved in the SKIZ definition have been generalized to fuzzy sets (such as distances, dilations, erosions) influence zones and SKIZ have, to the best of our knowledge, never been defined in the case of fuzzy sets. This is the aim of this section. We consider two fuzzy sets, with membership functions μ_1 and μ_2 defined on $\mathcal{S}$. The extension to an arbitrary number of fuzzy sets is then straightforward.

2.1 Fuzzy structuring element and fuzzy dilation and erosion

The morphological operations involved in the crisp case are performed using a structuring element which is a ball of a distance. In $\mathbb{R}^n$, the Euclidean distance is generally considered. In a digital space, such as $\mathbb{Z}^n$, a discrete distance is defined, based on an underlying discrete connectivity. The ball of radius 1 of this distance is then constituted by the center point and its neighbors according to the choice of the connectivity. More generally, the structuring element can be defined from a binary relation on $\mathcal{S}$, that is assumed to be symmetrical in this paper (which is consistent if it is a ball of a distance). In the fuzzy case, the same crisp structuring elements can be used. We can also base the operations on a fuzzy structuring element, which can represent local imprecision or a fuzzy binary relation. We denote the structuring element by its membership function ν. All what follows applies for crisp and for fuzzy structuring elements. In $\mathbb{R}^n$ or $\mathbb{Z}^n$, $\nu(x)$ represents the degree to which x belongs to ν and $\nu(y - x)$ the degree to which y belongs to the translation of ν at point x. If ν is derived from a fuzzy binary relation, $\nu(y - x)$ denotes the degree to which y is in relation to x.

Let us denote by $\delta_\nu(\mu)$ and $\varepsilon_\nu(\mu)$ the dilation and erosion of the fuzzy set μ by the structuring element ν. Here, dual definitions of these operations are chosen [5], i.e. verifying $\varepsilon_\nu(\mu^c) = (\delta_\nu(\mu))^c$, since this property is important, as seen in Equation 2. They are expressed as:

$$\forall x \in \mathcal{S},\ \delta_\nu(\mu)(x) = \sup_{y \in \mathcal{S}} \top[\mu(y), \nu(y-x)], \tag{3}$$

$$\forall x \in \mathcal{S},\ \varepsilon_\nu(\mu)(x) = \inf_{y \in \mathcal{S}} \bot[\mu(y), c(\nu(y-x))], \tag{4}$$

where $\top$ is a t-norm and $\bot$ the t-conorm dual of $\top$ with respect to a complementation c (which automatically guarantees the duality between δ and ε). Examples of t-norms are min, product, Lukasiewicz ($\max(0, a+b-1)$), and they generalize intersection to fuzzy sets, while t-conorms generalize union (examples are max, algebraic sum and Lukasiewicz $\min(1, a+b)$). In this paper, the following classical complementation is used: $\forall t \in [0,1], c(t) = 1-t$. Other definitions of fuzzy mathematical morphology have been proposed (e.g. [7,8,14]), based on different operators. Links with the ones used here are developed in [3,5].

Important properties of the definitions given in Equations 3 and 4, that will be intensively used in the following, are: (i) fuzzy dilation and erosion are equivalent to the classical dilation and erosion in case both μ and ν are crisp; (ii) $\nu(0) = 1 \Rightarrow \mu \leq \delta_\nu(\mu)$ and $\varepsilon_\nu(\mu) \leq \mu$, where 0 denotes the origin of $\mathcal{S}$ (if ν represents a binary relation, it means that this relation is reflexive); (iii) fuzzy dilation and erosion are increasing with respect to μ, dilation is increasing with respect to ν while erosion is decreasing; (iv) fuzzy dilation commutes with the supremum and fuzzy erosion with the infimum; (v) duality: $\varepsilon_\nu(\mu^c) = (\delta_\nu(\mu))^c$; (vi) iterativity property: successive dilations (respectively erosions) are equivalent to one dilation (respectively erosion) with a structuring element equal to the dilation of all structuring elements.

2.2 Definition based on fuzzy dilations

Let us first consider the expression of influence zone using morphological dilations (Equation 2). This expression can be extended to fuzzy sets by using fuzzy intersection and union, and fuzzy mathematical morphology.

Definition 1. For a given structuring element ν, we define the influence zone of μ_1 as:

$$IZ_{dil}(\mu_1) = \bigcup_\lambda (\delta_{\lambda\nu}(\mu_1) \cap \varepsilon_{\lambda\nu}(\mu_2^c)) = \bigcup_\lambda (\delta_{\lambda\nu}(\mu_1) \setminus \delta_{\lambda\nu}(\mu_2)). \tag{5}$$

The dilation by $\lambda\nu$ is obtained by λ iterations of a dilation by ν in the discrete case. The influence zone for μ_2 is defined in a similar way. The extension to any number of fuzzy sets μ_i is straightforward: $IZ_{dil}(\mu_i) = \bigcup_\lambda (\delta_{\lambda\nu}(\mu_i) \cap \varepsilon_{\lambda\nu}((\cup_{j \neq i}\mu_j)^c))$. $\square$

In these equations, intersection and union of fuzzy sets are implemented as t-norms $\top$ and t-conorms $\bot$ (min and max for instance). Equation 5 then reads: $IZ_{dil}(\mu_1) = \sup_\lambda \top[\delta_{\lambda\nu}(\mu_1), 1 - \delta_{\lambda\nu}(\mu_2)]$.

Note that the number of dilations to be performed to compute influence zones in a digital bounded space $\mathcal{S}$ is always finite (and bounded by the length of the largest diagonal of $\mathcal{S}$).

Definition 2. The fuzzy SKIZ is then defined as:

$$\mathrm{SKIZ}(\cup_i \mu_i) = (\textstyle\bigcup_i IZ(\mu_i))^c.$$

$\square$

This expression also defines a fuzzy (generalized) Voronoï diagram. Although the notion of Voronoï diagram has already been used in fuzzy systems, to our knowledge, no fuzzy version of it was defined until now.

2.3 Definitions based on distances

Another approach consists in extending the definition in terms of distances (Equation 1) and defining a degree to which the distance to one of the sets is lower than the distance to the other sets. Several definitions of the distance of a point to a fuzzy set have been proposed in the literature. Some of them provide real numbers and Equation 1 can then be applied directly. But then the imprecision in the object definition is lost. Definitions providing fuzzy numbers are therefore more interesting, since if the sets are imprecise, it may be expected that distances are imprecise too, as also underlined e.g. in [2, 10]. In particular, as will be seen next, it may be interesting to use the distance proposed in [2], based on fuzzy dilation:

$$d(x, \mu)(n) = \top[\delta_{n\nu}(\mu)(x), 1 - \delta_{(n-1)\nu}(\mu)(x)]. \tag{6}$$

It expresses, in the digital case, the degree to which x is at a distance n of μ ($\top$ is a t-norm, and $n \in \mathbb{N}^*$). For $n = 0$, the degree becomes $d(x, \mu)(0) = \mu(x)$. This expression can be generalized to the continuous case as:

$$d(x, \mu)(\lambda) = \inf_{\lambda' < \lambda} \top[\delta_{\lambda\nu}(\mu)(x), 1 - \delta_{\lambda'\nu}(\mu)(x)], \tag{7}$$

where $\lambda \in \mathbb{R}^{+*}$, and $d(x, \mu)(0) = \mu(x)$.

First method: comparing fuzzy numbers

When distances are fuzzy numbers, the fact that $d(x, \mu_1)$ is lower than $d(x, \mu_2)$ becomes a matter of degree. The degree to which this relation is satisfied can be performed using methods for comparing fuzzy numbers. Let us consider the definition in [9], which expresses the degree $\mu(d_1 < d_2)$ to which $d_1 < d_2$, d_1 and d_2 being two fuzzy numbers, using the extension principle:

$$\mu(d_1 < d_2) = \sup_{a<b} \min(d_1(a), d_2(b)). \tag{8}$$

Definition 3. The influence zone of μ_1 based on the comparison of fuzzy numbers (using Equation 8) is defined as:

$$\begin{aligned} IZ_{dist1}(\mu_1)(x) &= \mu(d(x,\mu_1) < d(x,\mu_2)) \\ &= \sup_{n<n'} \min[d(x,\mu_1)(n), d(x,\mu_2)(n')]. \end{aligned} \tag{9}$$

□

Note that this approach can be applied whatever the chosen definition of fuzzy distances.

Second method: direct approach

When distances are more specifically derived from a dilation, as the ones in Equations 6 and 7, a more direct approach can be proposed, taking into account explicitly this link between distances and dilations. Indeed, in the binary case, the following equivalences hold:

$$\begin{aligned} (d(x,X_1) \leq d(x,X_2)) &\Leftrightarrow (\forall\lambda, x \in \delta_\lambda(X_2) \Rightarrow x \in \delta_\lambda(X_1)) \\ &\Leftrightarrow (\forall\lambda, x \in \delta_\lambda(X_1) \vee x \notin \delta_\lambda(X_2)). \end{aligned} \tag{10}$$

This expression extends to the fuzzy case as follows.

Definition 4. The degree $\mu(d(x,\mu_1) \leq d(x,\mu_2))$ to which $d(x,\mu_1)$ is less than $d(x,\mu_2)$ is defined as:

$$\mu(d(x,\mu_1) \leq d(x,\mu_2)) = \inf_\lambda \bot(\delta_{\lambda\nu}(\mu_1)(x), 1 - \delta_{\lambda\nu}(\mu_2)(x)), \tag{11}$$

where $\bot$ is a t-conorm. □

This equation defines a new way to compare fuzzy numbers representing distances.

The comparison of fuzzy numbers representing distances, as given by Equation 11 is reflexive ($\mu(d(x,\mu_1) \leq d(x,\mu_1)) = 1$) if and only if $\bot$ is a t-conorm verifying the excluded middle law (Lukasiewicz t-conorm for instance). Moreover, in case the fuzzy numbers are usual numbers, the comparison reduces to the classical comparison between numbers.

Defining influence zones requires a strict inequality between distances, which is deduced by complementation:

$$\mu(d(x,\mu_1) < d(x,\mu_2)) = 1 - \mu(d(x,\mu_2) \leq d(x,\mu_1)). \tag{12}$$

Definition 5. The influence zone of μ_1 using the comparison introduced in Definition 4 is defined as:

$$IZ_{dist2}(\mu_1)(x) = 1 - \inf_\lambda \bot(\delta_{\lambda\nu}(\mu_2)(x), 1 - \delta_{\lambda\nu}(\mu_1)(x)). \tag{13}$$

□

Whatever the chosen definition of IZ, the SKIZ is always defined as in Definition 2.

2.4 Comparison and properties

Proposition 1. *Definitions 1 and 5 are equivalent:* $IZ_{dil}(\mu_1) = IZ_{dist2}(\mu_1)$.

Although this result is not surprising, both interpretations in terms of dilation and distance remain interesting.

However, the two distance based approaches are not equivalent, since they rely on different orderings between fuzzy sets. Actually the direct approach always provides a larger result.

Proposition 2. $\forall x \in \mathcal{S}, IZ_{dist1}(\mu_1)(x) \leq IZ_{dist2}(\mu_1)(x)$.

Proposition 3. *For* N *being the size of* $\mathcal{S}$ *in each dimension, the complexity of computation of* IZ_{dist1} *is in* $O(N^5)$ *in 3D and* $O(N^4)$ *in 2D. The complexity of computation of* IZ_{dist2} *or* IZ_{dil} *is at least one order of magnitude less.*

Proposition 4. *Definitions 1, 2, 3 and 5 are equivalent to the classical definitions in case of crisp sets and crisp structuring elements.*

Finally, the SKIZ is symmetrical with respect to the μ_i, hence independent of their order.

2.5 Illustrative example

The notion of fuzzy SKIZ is illustrated on the three objects of Figure 1. The structuring element ν is a crisp 3×3 square in Figure 2 and a fuzzy set of paraboloid shape in Figure 3. The influence zones of each object are displayed, as well as the SKIZ. These results are obtained with the dilation based definition. Each influence zone is characterized by high membership values close to the corresponding object, and decreasing when the distance to this object increases. The use of a fuzzy structuring element results in more fuzziness in the influence zones and SKIZ.

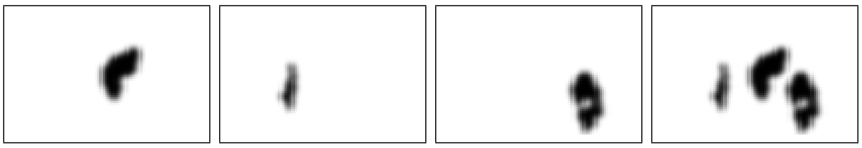


Figure 1. Three fuzzy objects and their union. Membership degrees range from 0 (white) to 1 (black).

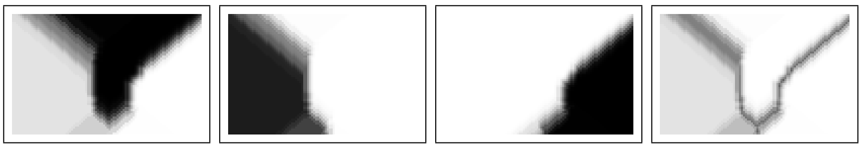


Figure 2. Influence zones of the three fuzzy objects of Figure 1 and resulting fuzzy SKIZ, obtained using a binary structuring element (3×3 square).

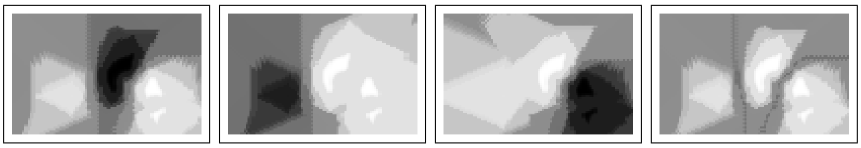


Figure 3. Influence zones of the three fuzzy objects of Figure 1 and resulting fuzzy SKIZ, obtained using a fuzzy structuring element (paraboloid shaped).

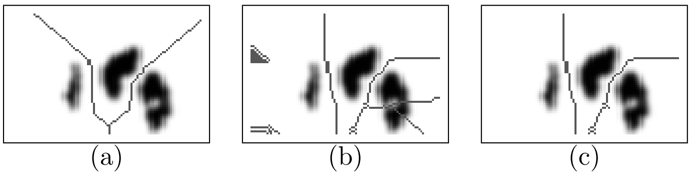


Figure 4. Binary decision using watershed for ν crisp (a) and fuzzy (b). Lines with a very low membership degree in the SKIZ of (b) have been suppressed in (c).

A binary decision can be made in order to obtain a crisp SKIZ of fuzzy objects. An appropriate approach consists in computing the watershed lines of the fuzzy SKIZ. It is appropriate in the sense that it provides spatially consistent lines, without holes, and going through the crest lines of the membership function of the SKIZ. A result is provided in Figure 4. For a fuzzy structuring element ν, the lines can go through the objects (Figure 4(b)). While this is impossible in the binary case, in the fuzzy case this is explained by the fact that an object can, to some degree, be built of several connected components, linked together by points with low membership degrees. The values of the SKIZ at those points are low too. This is the case for the third object in Figure 1. The low values of the SKIZ along the line traversing this object are in accordance with the fact that the object has only one connected component with some degree, and two components with some degree. The line separating the third object can be suppressed by eliminating the parts of the watersheds having a very low degree in the fuzzy SKIZ (Figure 4(c)). This requires to set a threshold value.

3. Fuzzy median set and interpolation between fuzzy sets

In the mathematical morphology community, two types of approaches have been considered to define the median set of two crisp sets, or to interpolate between two sets. The first one relies on the SKIZ [1, 19], while the second one relies on the notion of geodesics of some distance [11, 16, 17]. Here, we propose to extend the first approach to the case of fuzzy sets, based on the definitions of the fuzzy SKIZ proposed in Section 2. The median set of two intersecting sets X and Y is defined as the influence zone of $X_1 = X \cap Y$ with respect to $X_2 = (X \cup Y)^c$.

3.1 Definitions

Let us consider two fuzzy objects with membership functions μ_1 and μ_2 and with intersecting supports. Two definitions can be given for the fuzzy median set, depending on the chosen definition for the influence zones.

Definition 6. Based on the definition of influence zones from dilations, or equivalently the direct approach from distances, the median fuzzy set of μ_1 and μ_2 is defined as the influence zone of $\mu_1 \cap \mu_2$ with respect to $(\mu_1 \cup \mu_2)^c$ (intersection is still defined by a t-norm and union by a t-conorm):

$$\begin{aligned}\forall x \in \mathcal{S}, M(\mu_1, \mu_2)(x) &= \sup_{\lambda} \top[\delta_{\lambda\nu}(\mu_1 \cap \mu_2)(x), 1 - \delta_{\lambda\nu}((\mu_1 \cup \mu_2)^c)(x)] \\ &= \sup_{\lambda} \top[\delta_{\lambda\nu}(\mu_1 \cap \mu_2)(x), \varepsilon_{\lambda\nu}(\mu_1 \cup \mu_2)(x)]. \quad (14)\end{aligned}$$

□

Definition 7. By using the definition of influence zones based on comparison of fuzzy distances, the median set is defined as:

$$M'(\mu_1, \mu_2)(x) = \sup_{n<n'} \min[d(x, \mu_1 \cap \mu_2)(n), d(x, (\mu_1 \cup \mu_2)^c)(n')]. \quad (15)$$

□

Proposition 5. *For any two fuzzy sets μ_1 and μ_2, we always have:*

$$\forall x \in \mathcal{S}, M'(\mu_1, \mu_2)(x) \leq M(\mu_1, \mu_2)(x). \quad (16)$$

This notion of median set can be exploited to derive a series of interpolating sets between μ_1 and μ_2, by applying recursively the median computation in a dichotomic process.

Definition 8. Let μ_1 and μ_2 be two fuzzy sets. A series of interpolating sets is defined by recursive application of the median computation:

$$\begin{aligned} Interp_0 &= \mu_1 \quad Interp_1 = \mu_2 \\ Interp_{\frac{i+j}{2}} &= M(Interp_i, Interp_j) \text{ for } 0 \leq i \leq 1, 0 \leq j \leq 1. \end{aligned}$$

□

This sequence allows transforming progressively μ_1 into μ_2. These two fuzzy sets can represent spatial objects, different situations, sets of constraints or preferences, etc. For instance the sequence allows building intermediate estimates between distant observations or pieces of information.

3.2 Examples

On various examples, it can be actually observed that M' leads to lower membership values than M (Proposition 5). The result provided by M is visually more satisfactory and is moreover faster to compute. Therefore, in the following the chosen definition is the one given by Equation 14. Figure 5 illustrates an example of interpolation between two fuzzy sets. The series of interpolating fuzzy sets is computed recursively from the median set (the fourth set in the sequence displayed in the figure). It is clear on this example that the shape of interpolating sets evolves progressively from the one of the first object towards the one of the second object. This evolution is in accordance with the expected interpolation notion.

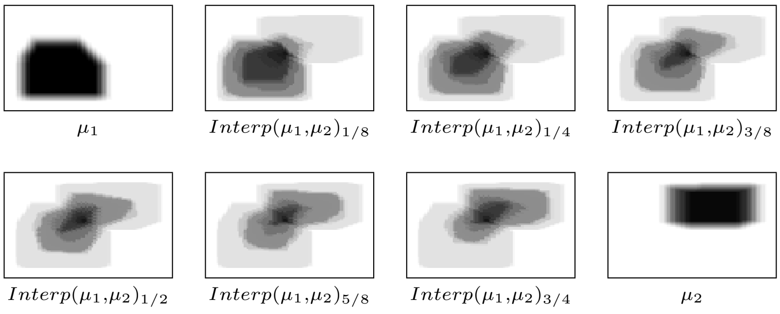

μ_1
$Interp(\mu_1,\mu_2)_{1/8}$
$Interp(\mu_1,\mu_2)_{1/4}$
$Interp(\mu_1,\mu_2)_{3/8}$
$Interp(\mu_1,\mu_2)_{1/2}$
$Interp(\mu_1,\mu_2)_{5/8}$
$Interp(\mu_1,\mu_2)_{3/4}$
μ_2

Figure 5. Interpolation between two fuzzy sets μ_1 and μ_2 ($Interp(\mu_1, \mu_2)_{1/2} = M(\mu_1, \mu_2)$).

Let us now consider real objects, from medical images. We consider the putamen (a brain structure) in different subjects, obtained from the IBSR database[1]. The images are registered, which guarantees a good correspondence between the different instances. Fuzziness at the boundary of the objects is introduced to represent spatial imprecision due to partial volume effect or imprecise segmentation, using a fuzzy dilation. Four examples of the resulting fuzzy objects are illustrated in Figure 6. The fuzzy median set has been computed between the two first instances, then between this result and the third instance, etc. Results are displayed in Figure 6. Using this iterative approach, the fuzzy median set between the 18 instances of this structure has been computed (corresponding to the 18 normal subjects of the IBSR database). Such results could be used for instance for representing the inter-individual variability, or to build anatomical atlases.

Let us now consider another example, in the domain of preference modeling, as in [13], on which morphological operators can be defined [4]. We consider a propositional language based on a finite set of propositional symbols, on which formulas are defined. We denote by Ω the set of all interpretations. The models of a formula are considered as a fuzzy subset of Ω. To illustrate the application of the median operator, we consider a simple example, with three propositional symbols a, b, c, and two formulas φ_1 and φ_2, expressing respectively preferences for $\neg ab\neg c$ with a degree 0.2, and preferences for anything except abc with degrees as given in Table 1. For defining the morphological operators, we use the Hamming distance (i.e. two models are at a distance equal to the number of symbols instantiated

[1]http://www.cma.mgh.harvard.edu/ibsr/

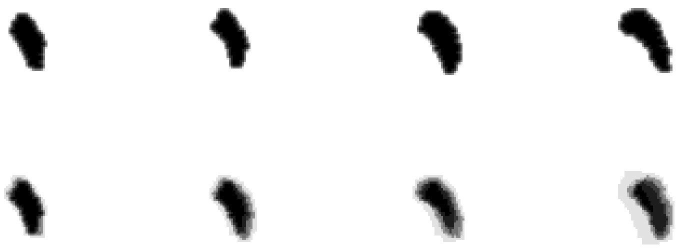


Figure 6. Four instances of a brain structure from four difference subjects, and median set between two, three, four instances and between the 18 instances of the IBSR database.

differently), and the structuring elements are the balls of this distance. The conjunction of φ_1 and φ_2 is equal to φ_1 and their disjunction is equal to φ_2.

Table 1. Fuzzy sets of Ω representing the preferences expressed by φ_1 and φ_2, and derivation of $M(\varphi_1, \varphi_2)$.

Models	abc	$\neg abc$	$a\neg bc$	$ab\neg c$	$\neg a\neg bc$	$\neg ab\neg c$	$a\neg b\neg c$	$\neg a\neg b\neg c$
φ_1	0	0	0	0	0	0.2	0	0
φ_2	0	0.5	0.5	0.5	0.5	0.8	0.5	0.7
$\delta_1(\varphi_1)$	0	0.2	0	0.2	0	0.2	0	0
$\varepsilon_1(\varphi_2)$	0	0	0	0	0.5	0.5	0.5	0.5
$\delta_2(\varphi_1)$	0.2	0.2	0	0.2	0.2	0.2	0.2	0.2
$\varepsilon_2(\varphi_2)$	0	0	0	0	0	0	0	0.5
$M(\varphi_1, \varphi_2)$	0	0	0	0	0	0.2	0	0.2

The successive steps of the computation of the median set are illustrated in Table 1. The models of the median set also constitute a fuzzy set of Ω. On this example, the classical fusion, according to [12] would lead to the intersection of the sets of models, i.e. φ_1. The result of the median is somewhat larger, since it includes also a model of φ_2 that was not a model of φ_1 ($\neg a\neg b\neg c$), and gives a more fair point of view expressing an intermediate solution between both sets of preferences. This can be interpreted as follows: if an individual as a set of preferences described by φ_1, which is very strict and constraining, he will be tempted to extend his preferences to obtain a better agreement with the preferences of the second individual. On the other hand, the second individual is ready to restrict his choices to achieve a consensus with the first one, and will be more satisfied if a fair account of all his preferences is obtained. Note that the fact that the median is included in the disjunction (see Proposition 6) guarantees that it does not contain a solution that nobody wants to accept. The resulting membership degrees reflect the low consistency that exists between both sets of preferences on this example.

3.3 Some properties

Proposition 6. *If $\nu(0) = 1$ (or if ν represents a reflexive relation), then the median set is included in the union of the two objects: $\forall x \in \mathcal{S}, M(\mu_1, \mu_2)(x) \leq (\mu_1 \cup \mu_2)(x)$.*

Proposition 7. *Under the same condition ($\nu(0) = 1$), the cores verify the following inclusion relations: $Core(\mu_1 \cap \mu_2) \subseteq Core(M(\mu_1, \mu_2)) \subseteq Core(\mu_1 \cup \mu_2)$.*

The core of a fuzzy set is the set of points having a membership value equal to 1. Note that the core of the median set can be empty.

Proposition 8. *If additionally the origin is the only modal value of ν ($\nu(0) = 1$ and $\forall x \in \mathcal{S} \setminus \{0\}, \nu(x) < 1$), then the median set and the union of the two sets have the same support and the cores of the median set and of the intersection are equal.*

In particular, in the case where the structuring element is crisp (for instance a square of size 3×3), this property does not hold, while it holds for the paraboloid shaped structuring element used in the presented results.

Figure 5 illustrates that the median set and the union have the same support. It should be noticed that in a large part of the support, the membership values are very low (0.1 in this example), and that it would be very easy to eliminate these low values if a more reduced support is desired, as could be intuitively preferable.

4. Conclusion

In this paper, novel notions of fuzzy SKIZ, median and interpolation were introduced, based on mathematical morphology concepts. The proposed definitions are applicable whatever the dimension of the underlying space $\mathcal{S}$ and whatever the semantics attached to the fuzzy sets. The only hypothesis is that it should be possible to define a structuring element, either from a distance on $\mathcal{S}$, or from a binary symmetrical relation.

The definitions of median set and interpolation can be extended to non intersecting fuzzy sets if a translation on $\mathcal{S}$ can be defined. The cases where $\mathcal{S}$ does not have an affine structure are planned for future work.

Another approach for defining median sets in the crisp case is based on geodesic distances [16]. Extension of this approach to the fuzzy case could be another interesting research direction. Extensions to a logical framework for mediation applications was proposed in [6], but not to the fuzzy sets framework until now.

Extensions of the median set to more than two fuzzy sets could be interesting too, for instance for deriving generic models from different instances.

In the brain structure example, the median has been computed iteratively, a process which depends on the order. A direct method involving all objects simultaneously deserves to be developed.

Finally, applications of the propositions of this paper could be further explored, for instance for fusion, with a comparison to other operators also based on distance [12, 13], for the definition of compromises or negotiations, for smoothing fuzzy sets representing preferences, observations, etc., or for finding the fuzzy sets in between two sets.

References

[1] S. Beucher, *Sets, Partitions and Functions Interpolations*, ISMM'98, 1998, pp. 307–314.

[2] I. Bloch, *On Fuzzy Distances and their Use in Image Processing under Imprecision*, Pattern Recognition **32** (1999), no. 11, 1873–1895.

[3] ______, *Duality vs Adjunction and General Form for Fuzzy Mathematical Morphology*, WILF, 2005, pp. 354–361.

[4] I. Bloch and J. Lang, *Towards Mathematical Morpho-Logics*, 8th International Conference on Information Processing and Management of Uncertainty in Knowledge based Systems IPMU 2000, 2000, pp. 1405–1412.

[5] I. Bloch and H. Maître, *Fuzzy Mathematical Morphologies: A Comparative Study*, Pattern Recognition **28** (1995), no. 9, 1341–1387.

[6] I. Bloch, R. Pino-Pérez, and C. Uzcategui, *Mediation in the Framework of Morphologic*, European Conference on Artificial Intelligence ECAI 2006, 2006, pp. 190–194.

[7] B. de Baets, *Fuzzy Morphology: a Logical Approach*, Uncertainty in Engineering and Sciences: Fuzzy Logic, Statistics and Neural Network Approach, 1997, pp. 53–67.

[8] T.-Q. Deng and H. J. A. M. Heijmans, *Grey-Scale Morphology Based on Fuzzy Logic*, Journal of Mathematical Imaging and Vision **16** (2002), 155–171.

[9] D. Dubois and H. Prade, *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, New-York, 1980.

[10] ______, *On Distance between Fuzzy Points and their Use for Plausible Reasoning*, Int. Conf. Systems, Man, and Cybernetics, 1983, pp. 300–303.

[11] I. Granado, N. Sirakov, and F. Muge, *A Morphological Interpolation Approach - Geodesic Set Definition in Case of Empty Intersection*, ISMM'00, 2000, pp. 71–80.

[12] S. Konieczny and R. Pino-Pérez, *Merging Information: A Qualitative Framework*, Journal of Logic and Computation **12** (2002), no. 5, 773–808.

[13] C. Lafage and J. Lang, *Logical Representation of Preferences for Group Decision Making*, 7th International Conference on Principles of Knowledge Representation and Reasoning KR 2000, 2000, pp. 457–468.

[14] M. Nachtegael and E. E. Kerre, *Classical and Fuzzy Approaches towards Mathematical Morphology*, Fuzzy Techniques in Image Processing, 2000, pp. 3–57.

[15] J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, London, 1982.

[16] J. Serra, *Hausdorff Distances and Interpolations*, ISMM'98, 1998, pp. 107–114.

[17] P. Soille, *Generalized Geodesic Distances Applied to Interpolation and Shape Description*, ISMM'94, 1994, pp. 193–200.

[18] ______, *Morphological Image Analysis: Principles and Applications*, Springer-Verlag, Berlin, 1999.

[19] J. Vidal, J. Crespo, and V. Maojo, *Recursive Interpolation Technique For Binary Images Based on Morphological Median Sets*, ISMM'05, 2005, pp. 53–62.

Basis Computation Algorithms

Nina S. T. Hirata, Roberto Hirata Jr. and Junior Barrera

Instituto de Matemática e Estatística (IME), Universidade de São Paulo (USP), SP, Brazil
{nina,hirata,jb}@ime.usp.br

Abstract Algorithms for the computation of the basis (maximal intervals of the kernel) of binary and gray-scale translation-invariant and locally defined morphological operators are presented. Some examples that illustrate the dynamics of the algorithms as well as their use as learning algorithms are also presented.

Keywords: kernel, basis, maximal intervals, incremental splitting of intervals.

1. Introduction

One interesting aspect of morphological operators is the existence of canonical representations [1, 2, 9–11]. According to the canonical decomposition theorem, any translation-invariant morphological operator can be expressed as a supremum of sup-generating (also known as interval) operators. Although this theorem holds for any translation-invariant mapping between two complete lattices, in this work we restrict the scope to binary and gray-level image operators. If we also impose local definition by a neighborhood W, it can be shown that all kernel elements can be restricted to W [5]. Such operators are called W-operators. A W-operator can be characterized by a function whose domain is restricted to sub-images in W and its value to gray-scales of the image. This fact is important if our concern is the computational representation of image operators.

In the binary case, a W-operator $\Psi : \mathcal{P}(E) \to \mathcal{P}(E)$, where $E = \mathbb{Z}^2$ denotes the image domain, is characterized by a binary function $\psi : \mathcal{P}(W) \to \{0, 1\}$. Its kernel is defined as

$$\mathcal{K}(\Psi) = \{X \in \mathcal{P}(W) : \psi(X) = 1\}, \tag{1}$$

and its basis, denoted $\mathbf{B}(\Psi)$, consists of the set of maximal intervals in the kernel.

Interval operators are parameterized by a pair of structuring elements that form intervals. The kernel induces a canonical decomposition while the basis induces a minimal decomposition as a supremum of interval operators. In terms of its kernel and basis, ψ can be expressed respectively as follows:

$$\psi(X) = max\{\lambda_{[A,A]}(X) : [A, A] \subseteq \mathcal{K}(\psi)\}\,, \tag{2}$$

and

$$\psi(X) = max\{\lambda_{[A,B]}(X) : [A,B] \in \mathbf{B}(\psi)\}\,, \tag{3}$$

where the interval operator $\lambda_{[A,B]}$ is defined as

$$\lambda_{[A,B]}(X) = 1 \Longleftrightarrow X \in [A,B]\,. \tag{4}$$

Similar definitions exist for gray-scale W-operators [3].

In this work we are particularly concerned with a specific problem. Suppose we have knowledge of some elements that are and that are not elements of a kernel. How do we compute a set of intervals that are consistent with this set of elements? In other words, elements known to be in the kernel must belong to at least one of the intervals and those known not to be in the kernel must not belong to any of the intervals. By representing the set of elements by intervals, our goal is to have a compact representation of the operator. In particular, we are interested in the representation of functions of the following types:

a) $\psi : \{0,1\}^n \to \{0,1\}$, $n > 0$ an integer,

b) $\psi : \{0,1\}^n \to \{0,1,2,\ldots,k\}$, $n, k > 0$ integers, and

c) $\psi : \{0,1,2,\ldots,k_1\}^n \to \{0,1,\ldots,k_2\}$, $n, k_1, k_2 > 0$ integers.

These functions characterize, respectively, W-operators (a) between binary images, (b) between binary and gray-level images, and (c) between gray-level images.

In the following three sections we present algorithms for the computation of a set of maximal intervals covering the elements that are known to be in the kernel and not covering the ones that are known not to be in the kernel, for the three cases above. In the last section we present the conclusions of this work.

2. Binary input and output

Finding maximal intervals of the kernel of a binary W-operator is equivalent to finding the minimal sum of products form of a Boolean function. Classical algorithms such as the one due to Quine and McCluskey are well known in the fields of switching theory and logic design [6]. It is known that this problem is NP (computationally intractable). Therefore many implementations use heuristics to circumvent computational difficulties. One of such implementations is the Berkeley Espresso [4].

While Quine-McCluskey's approach follows a bottom-up strategy of joining smaller intervals in the kernel in order to generate a larger one, until no more joining are possible, another alternative option we have proposed is based on a top-down approach [8]. The approach, called ISI (Incremental Splitting of Intervals), starts the process with the whole interval $[\emptyset, W]$ and then removes from this interval the elements that are not in the kernel. The result of the removing operation can be expressed by means of a set of sub-intervals of the initial interval. Thus, subsequent removing have the effect of sequentially breaking (or splitting) one ore more of these intervals. At the end of the process, when all elements that are not in the kernel have been removed, only elements of the kernel will remain covered by the resulting intervals.

2.1 Algorithm review

The removing operation or splitting rule [8] can be thought as the difference between two intervals $[A, B]$ and $[C, D]$ expressed in terms of the maximal intervals contained in $[A, B] \setminus [C, D]$.

Proposition 1 (Splitting rule)**.** *Let $[A, B]$ and $[C, D]$ be two intervals in $[\emptyset, W]$. Then, the set of maximal intervals contained in $[A, B] \setminus [C, D]$ is given by*

$$\big\{[A, B \cap \{c\}^c] : c \in C \cap A^c\big\} \cup \big\{[A \cup \{d\}, B] : d \in D^c \cap B\big\}, \tag{5}$$

where the complement $\cdot^c$ is with relation to W. If $C = D = X$, $X \in \mathcal{P}(W)$, the rule simplifies to

$$\{[A, B \cap \{a\}^c] : a \in X \cap A^c\} \cup \{[A \cup \{b\}, B] : b \in B \cap X^c\}. \tag{6}$$

The ISI algorithm consists basically on applying the splitting rule repeatedly in order to remove from interval $[\emptyset, W]$ all elements X such that $\psi(X) = 0$ (those elements that are not in the kernel of ψ). Thus, after the removing process finishes, the remaining intervals cover the elements X such that $\psi(X) = 1$ and eventually some elements X such that $\psi(X)$ is unknown (don't cares). If $\{X : \psi(X) = 1\} \cup \{X : \psi(X) = 0\} = \mathcal{P}(W)$, then the resulting collection of intervals corresponds to the basis of ψ. Otherwise, it is a basis that is consistent with ψ with respect to the elements in $\{X : \psi(X) = 1\} \cup \{X : \psi(X) = 0\}$.

The collection of maximal intervals resulting from the removing process may contain redundant intervals. That is, it may contain an interval $[A, B]$ such that all its elements are contained in some other intervals of the collection (although $[A, B]$ itself is not contained in any other interval). Hence, in a second step of the ISI algorithm a minimum cover (sub-basis) should be computed [6]. A minimum cover corresponds to a smallest sub-collection of intervals that is enough to cover all elements in $\{X : \psi(X) = 1\}$.

When don't cares are present, some heuristics can be applied to accelerate the algorithm. For instance, during the removing process, intervals that do not contain any element of $\{X : \psi(X) = 1\}$ can be disregarded because they will not be needed for the minimum cover. As an example, in Figure 1, filled circles represent the elements of the kernel, non-filled circles are the elements for which ψ takes value 0, and the absence of a circle indicates an element for which the value of ψ is not defined (dont't care). Figure 1(a) shows the removing of element 001 from the interval $[000, 111]$, resulting in three intervals, $[000, 110]$, $[100, 111]$ and $[010, 111]$. The one (enclosed by a dashed box) does not contain any element of $\mathcal{K}(\psi)$ and, therefore, it can be discarded, as illustrated in Figure 1(b).

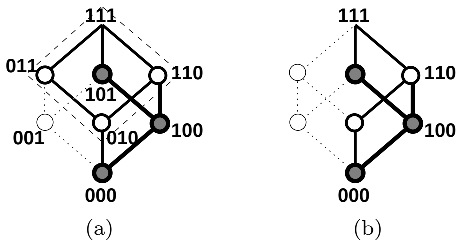


Figure 1. (a) The three intervals after the removing of 001. (b) Discarding of interval $[010, 111]$, which will not be needed for the minimum cover.

More details of this algorithm, as well as some interesting heuristics to improve the processing time, can be found in [8].

2.2 Application example

The above algorithm may be used as a learning (generalization) algorithm, as shown in the next example. Figure 2(a) shows an input-output pair of training images (to be used to estimate a kernel). From the training images,

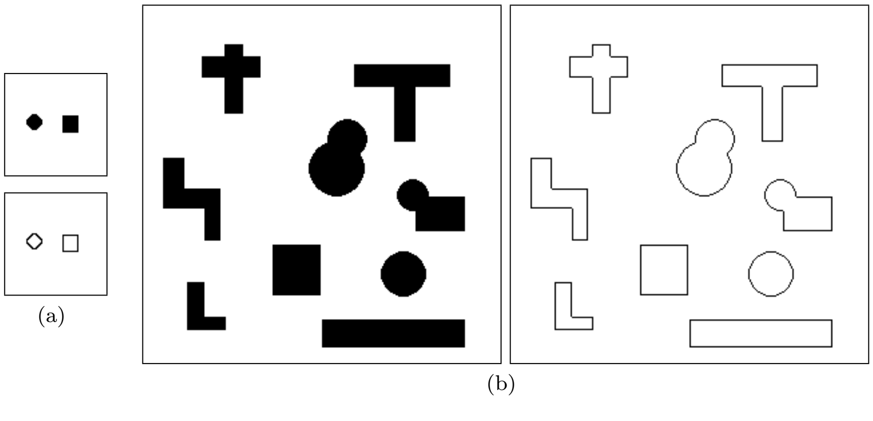
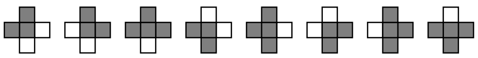
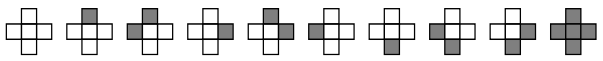


Figure 2. Learning binary image operators: (a) training images and (b) test images.

the elements known to be in the kernel are



while those known not to be in the kernel are



The resulting basis is given by the set of intervals

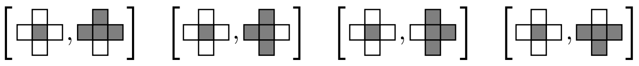


Figure 2(b) shows the result obtained by applying the above basis on another image. The learned operator is the 8-connected internal edge extractor. Additional application examples can be found in [7].

3. Binary input and multi-level output

Operators that map binary images to gray-scale images can be characterized by a function of the form $\psi : \{0,1\}^n \to \{0,1,\ldots,k\}$, where k denotes the maximum gray-level. Their kernel are defined by level, that is, the *kernel of ψ at level i*, $i = 0,1,\ldots,k$, is the collection $\mathcal{K}_i(\psi)$ given by, for any $\mathbf{x} \in \{0,1\}^n$,

$$\mathbf{x} \in \mathcal{K}_i(\psi) \iff \psi(\mathbf{x}) \geq i. \tag{7}$$

Notice that $\mathcal{K}_k(\psi) \subseteq \mathcal{K}_{k-1}(\psi) \subseteq \cdots \subseteq \mathcal{K}_1(\psi) \subseteq \mathcal{K}_0(\psi)$. The *basis of ψ at level i*, $\mathbf{B}_i(\psi)$, is the collection of maximal intervals in $\mathcal{K}_i(\psi)$.

The sup-decomposition [3] of ψ in terms of its kernel and basis are given, respectively, $\forall \mathbf{x} \in \{0,1\}^n$, by

$$\psi(\mathbf{x}) = max\{i : \mathbf{x} \in \mathcal{K}_i(\psi), i = 0,1,\ldots,k\}, \tag{8}$$

and

$$\psi(\mathbf{x}) = max\{i : \mathbf{x} \in [\mathbf{a},\mathbf{b}], [\mathbf{a},\mathbf{b}] \in \mathbf{B}_i(\psi), i = 0,1,\ldots,k\}\,. \tag{9}$$

3.1 Algorithm

To compute the basis $\mathbf{B}_i(\psi)$, $i = 0,1,\ldots,k$, of $\psi : \{0,1\}^n \to \{0,1,\ldots,k\}$, ISI is applied repeatedly, one time for each level i. Initially, all elements with label 0 are removed from the interval $[\emptyset, W]$. In this process, all other elements are regarded as 1s. The resulting intervals correspond to $\mathbf{B}_1(\psi)$, the intervals that cover all elements with label greater or equal to 1. In the next step, the same process is repeated for all elements with label 1. This time, the initial intervals are those in $\mathbf{B}_1(\psi)$ and the resulting intervals correspond to $\mathbf{B}_2(\psi)$. This process is repeated successively for the next labels until all elements of label $k-1$ are removed, resulting in the basis at level k.

As an example, consider $\psi : \mathcal{P}(W) \to \{0,1,2,3\}$, with $|W| = 3$, given by $\psi(000) = 0$, $\psi(010) = 1$, $\psi(100) = 1$, $\psi(101) = 2$, $\psi(110) = 2$ and $\psi(111) = 3$. The remaining elements are don't cares. Figure 3 shows the dynamics of the algorithm, while Figure 4 shows the resulting basis. The basis at level 0, $\mathbf{B}_0(\psi)$, is the set composed by the interval $[\emptyset, W]$. To compute $\mathbf{B}_1(\psi)$, all elements with label 0 are removed from the interval $[\emptyset, W]$, resulting in $\mathbf{B}_1(\psi) = \{[001, 111], [010, 111], [100, 111]\}$. To compute $\mathbf{B}_2(\psi)$, all elements with label 1 are removed from the intervals in $\mathbf{B}_1(\psi)$, resulting in $\mathbf{B}_2(\psi) = \{[001, 111], [110, 111]\}$. Proceeding similarly, we obtain $\mathbf{B}_3(\psi) = \{[011, 111]\}$. Notice that intervals $X11$, $11X$, $1X1$ and 111 are discarded because they are contained in another intervals, while $0X1$ is discarded because it contains only don't cares.

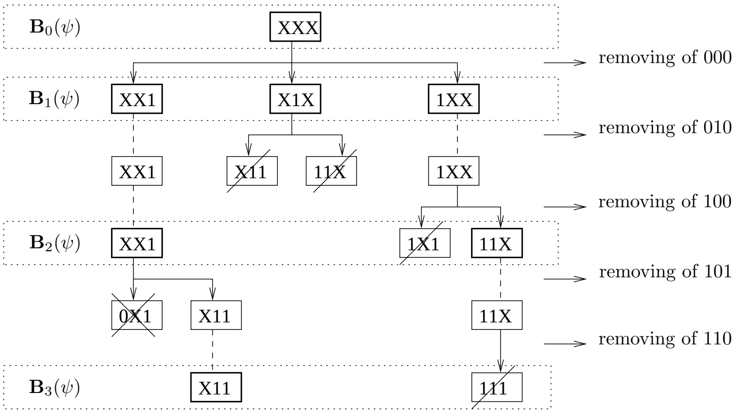


Figure 3. Example of the application of multi-level output ISI.

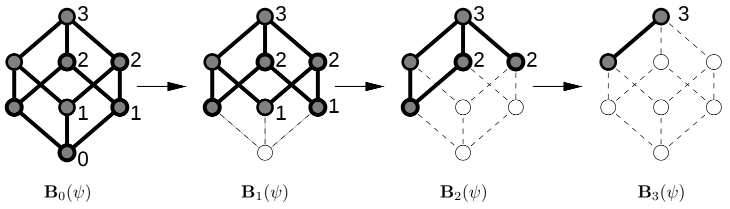


Figure 4. Lattice view of the intervals of Figure 3.

Given a partially defined ψ, let $\mathcal{M} = \{(X, l) : \psi(X) = l, l = 0, 1, \ldots, k\}$ (all elements whose label is known). The basis computation process is summarized in Algorithm 1.

4. Gray-level input and output

Gray-scale W-operators can be characterized by functions in the form $\psi : \{0, 1, \ldots, k_1\}^n \to \{0, 1, \ldots, k_2\}$, where k_1 and k_2 correspond to the input and output maximum gray-levels. Analogous to the previous case, the *kernel*

Algorithm 1 Basis computation.

```
1: $\mathbb{I} \leftarrow \{[\emptyset, W]\}$, with label 0; $\mathbb{I}_{\text{final}} \leftarrow \emptyset$ ;
2: $\mathcal{M}_1 = \mathcal{M}$ ;
   // Repeat for each level, except $k$
3: for all $l \in \{0, 1, \ldots, k-1\}$ do
4:   $\mathcal{M}_0 = \{(X, i) \in \mathcal{M}_1 : i = l\}$ ; // elements to be removed
5:   $\mathcal{M}_1 = \mathcal{M}_1 \setminus \mathcal{M}_0$ ; // elements to be covered
     // Remove each $X \in \mathcal{M}_0$ from $\mathbb{I}$
6:   for all $(X, l) \in \mathcal{M}_0$ do
7:     $\mathbb{I}_{\text{New}} \leftarrow \emptyset$ ;
8:     $\mathbb{I}_{\text{P}} \leftarrow \{[A, B] \in \mathbb{I} : X \notin [A, B]\}$ ;
9:     $\mathbb{I}_{\text{tmp}} \leftarrow \{[A, B] \in \mathbb{I} : X \in [A, B]\}$ ;
       // Split each interval that contains $X$
10:    for all $[A, B] \in \mathbb{I}_{\text{tmp}}$ do
11:      $\mathbb{I}_{\text{split}} \leftarrow$ maximal intervals in $[A, B] \setminus \{X\}$ ;
         // Discard all intervals that does not cover any element in $\mathcal{M}_1$ and all
         those that are covered by another interval
12:      for all $[A', B'] \in \mathbb{I}_{\text{split}}$ do
13:        if $\exists X \in \mathcal{M}_1$ such that $X \in [A', B']$ then
14:          if $[A', B'] \not\subseteq [E, F], \forall [E, F] \in \mathbb{I}_{\text{P}}$ then
15:            $\mathbb{I}_{\text{New}} \leftarrow \mathbb{I}_{\text{New}} \cup \{[A', B']\}$ ;
16:          end if
17:        end if
18:      end for
19:    end for
20:    $\mathbb{I} \leftarrow \mathbb{I}_{\text{P}} \cup \mathbb{I}_{\text{New}}$ ;
21:  end for
22:  Label all intervals in $\mathbb{I}$ with $l + 1$ ;
23:  $\mathbb{I}_{\text{final}} = \mathbb{I}_{\text{final}} \cup \mathbb{I}$ ;
24: end for
25: Return $\mathbb{I}_{\text{final}}$ ;
```

of ψ at level i, $i \in \{0, 1, \ldots, k_2\}$, is the collection $\mathcal{K}_i(\psi)$ given by, for any $\mathbf{x} \in \{0, 1, \ldots, k_1\}^n$,

$$\mathbf{x} \in \mathcal{K}_i(\psi) \Longleftrightarrow \psi(\mathbf{x}) \geq i, \tag{10}$$

and the *basis of ψ at level i*, $\mathbf{B}_i(\psi)$, is the collection of maximal intervals in $\mathcal{K}_i(\psi)$.

The sup-decomposition [3] of ψ in terms of kernel and basis are given, respectively, $\forall \mathbf{x} \in \{0, 1, \ldots, k_1\}^n$, by

$$\psi(\mathbf{x}) = max\{i \in \{0, 1, \ldots, k_2\} : \mathbf{x} \in \mathcal{K}_i(\psi)\}\,, \tag{11}$$

and

$$\psi(\mathbf{x}) = max\{i \in \{0, 1, \ldots, k_2\} : \mathbf{x} \in [\mathbf{a}, \mathbf{b}], [\mathbf{a}, \mathbf{b}] \in \mathbf{B}_i(\psi)\}\,. \tag{12}$$

In this section we present the extension of ISI for the computation of the basis. From the binary ISI we inherit the idea of splitting intervals, and from the multi-level output ISI we inherit the idea of a pyramid of basis.

4.1 Algorithm

To simplify notations, we consider $k_1 = k_2 = k$ and $K = \{0, 1, \ldots, k\}$, and in order to generalize the splitting rule for this case, we introduce two auxiliary functions ξ_I e ξ_O. Let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ be an element in K^W and let $O = (O_1, O_2, \ldots, O_n)$ and $I = (I_1, I_2, \ldots, I_n)$ be the smallest and the largest elements of K^W, respectively. We define

$$\xi_I(\mathbf{x}, i) = (I_1, \ldots, I_{i-1}, x_i - 1, I_{i+1}, \ldots, I_n) \tag{13}$$

and

$$\xi_O(\mathbf{x}, i) = (O_1, \ldots, O_{i-1}, x_i + 1, O_{i+1}, \ldots, O_n)\,. \tag{14}$$

Proposition 2. *Let* $[\mathbf{a}, \mathbf{b}]$ *and* $[\mathbf{c}, \mathbf{d}]$ *be two intervals in* K^W*, such that* $[\mathbf{c}, \mathbf{d}] \wedge [\mathbf{a}, \mathbf{b}] \neq \emptyset$*. Let* $\mathbf{a} = (a_1, a_2, \ldots, a_n)$*,* $\mathbf{b} = (b_1, b_2, \ldots, b_n)$*,* $\mathbf{c} = (c_1, c_2, \ldots, c_n)$ *and* $\mathbf{d} = (d_1, d_2, \ldots, d_n)$*, where* $a_i, b_i, c_i, d_i \in K, \forall i \in [1, n]$*. The set of maximal intervals resulting from the subtraction of* $[\mathbf{c}, \mathbf{d}]$ *from* $[\mathbf{a}, \mathbf{b}]$ *is given by*

$$\big\{[\mathbf{a}, \mathbf{b} \wedge \xi_I(\mathbf{c}, i)] : i \in [1, n]\big\} \cup \big\{[\mathbf{a} \vee \xi_O(\mathbf{d}, i), \mathbf{b}] : i \in [1, n]\big\}. \tag{15}$$

Similarly to the binary case, when $\mathbf{c} = \mathbf{d} = \mathbf{x}$, where $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, the above formula simplifies to

$$\big\{[\mathbf{a}, \mathbf{b} \wedge \xi_I(\mathbf{x}, i)] : i \in [1, n]\big\} \cup \big\{[\mathbf{a} \vee \xi_O(\mathbf{x}, i), \mathbf{b}] : i \in [1, n]\big\}. \tag{16}$$

After a removing operation, the number of resulting intervals as well as their dimension depend on the localization of the interval being removed. For an interval of dimension n, the number of resulting intervals may vary from n to $2n$. Figure 5 shows the three possibilities for the removing from an interval of dimension 2.

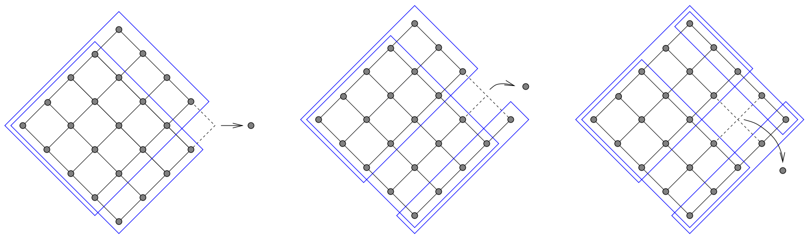


Figure 5. Three possible localizations of an element in an interval of dimension 2 and the intervals resulting from the splitting by that element.

Let $[A_1, B_1]$ and $[A_2, B_2]$ be two intervals such that there is a third interval $[C, D]$ that intercepts both (that is, $[C, D] \cap [A_1, B_1] \neq \emptyset$ and $[C, D] \cap [A_2, B_2] \neq \emptyset$). For the binary and multi-level output ISI, if we split both $[A_1, B_1]$ and $[A_2, B_2]$ by $[C, D]$, it is possible to show that no interval resulting from the splitting of $[A_1, B_1]$ is contained in any interval resulting from the splitting of $[A_2, B_2]$, and vice-versa [8].

However, for the gray-level ISI, the same is not true. For example, in Figure 6, two intervals $[10, 44]$ and $[04, 44]$ are given, and the element to be removed is 24. From the splitting of $[10, 44]$ there results three intervals, $[30, 44]$, $[10, 43]$ and $[10, 14]$. From the splitting of the second interval there results $[04, 14]$ and $[34, 44]$. Analyzing the resulting intervals, we observe that $[34, 44] \leq [30, 44]$. This implies that the algorithm must, besides verifying whether the resulting intervals of a splitting by $[\mathbf{c}, \mathbf{d}]$ are not contained in intervals of $\mathbb{I}_{\mathrm{P}}$ (those that do not intercept $[\mathbf{c}, \mathbf{d}]$), also verify if they are not contained in an interval resulting from the splitting of another interval by $[\mathbf{c}, \mathbf{d}]$.

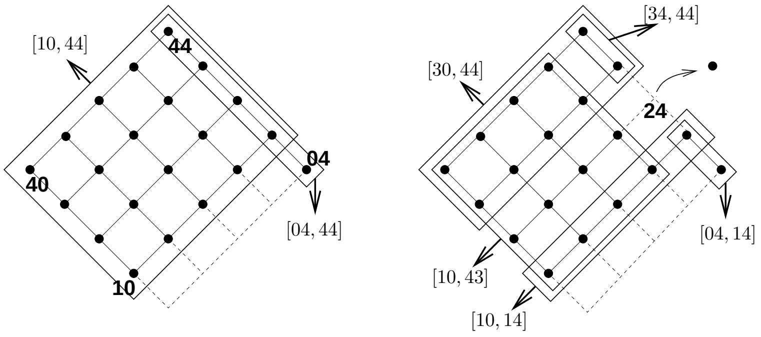


Figure 6. An interval resulting from the splitting of an interval may be contained in an interval resulting from the splitting of another interval by the same element.

As for the algorithm, the only modification needed with respect to the previously presented one is in Line 14, in order to include the verification discussed above. Of course, the splitting rule of Line 11 must be substituted by its equivalent for the gray-level case. Below follows Lines 14 to 16 of the modified algorithm, in order to compute the basis of a gray-level function.

```
if $[\mathbf{a}', \mathbf{b}'] \not\subseteq [\mathbf{e}, \mathbf{f}], \forall [\mathbf{e}, \mathbf{f}] \in \mathbb{I}_{\mathrm{P}} \cup \mathbb{I}_{\mathrm{New}}$ then
  $\mathbb{I}_{\mathrm{New}} \leftarrow \mathbb{I}_{\mathrm{New}} \cup \{[\mathbf{a}', \mathbf{b}']\}$ ;
end if
```

Figure 7 shows an example of the application of the gray-level ISI algorithm, step by step. The intervals obtained after removing each element are shown step by step in Figure 8. A minimum cover is computed each time all elements with a same label are removed.

Figures 8(b) to 8(e) show the intervals that result after removing elements with label 1, while Figures 8(f) to 8(h) show the intervals that result after removing elements with label 2. Intervals that do not cover any labeled elements are also removed and are not shown. Removing all elements with label 1 results in intervals that do not cover any element with label 1, but cover all elements of label greater than 1. Similarly, removing afterwards all

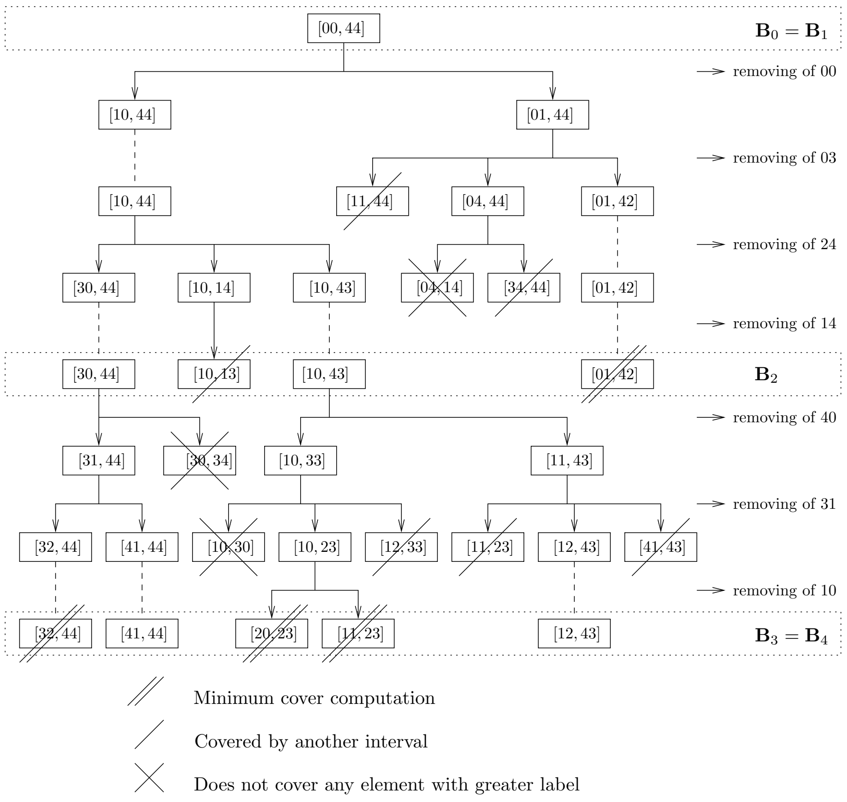


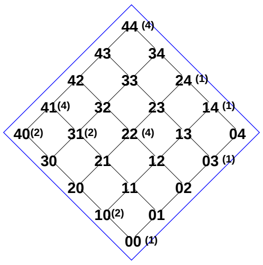
Figure 7. Example of the application of gray-level ISI.

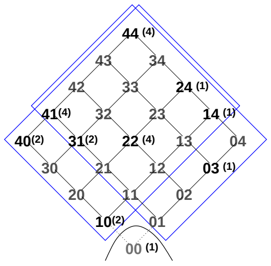
elements with label 2 results in intervals that cover all elements with label greater than 2, but do not cover any element with label 1 or 2.

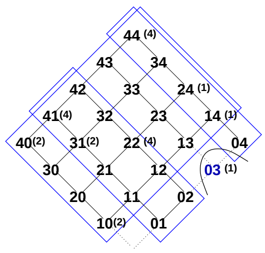
The basis at level 0 ($\mathbf{B}_0$) is the initial interval and since there are no elements with label 0, it is also the basis at level 1 ($\mathbf{B}_1$). Four and three splitting operations, respectively, were performed to remove elements with label 1 from $\mathbf{B}_1$ and elements with label 2 from $\mathbf{B}_2$. Since there are no elements with label 3, the basis at level 4 is the same at level 3. The resulting basis is depicted in Figure 9.

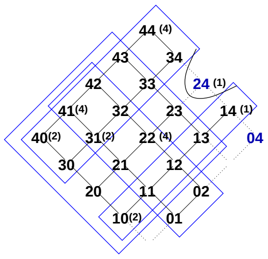
5. Conclusion

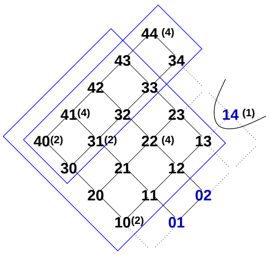
We have reviewed and presented algorithms for the computation of maximal intervals contained in the kernel of binary and gray-level morphological operators. They are based on operations that remove non-kernel elements from intervals and express the resulting elements of the interval by means of a set of subintervals. Implementation of these algorithms must take into consideration the fact that kernels are not always entirely known. Our al-

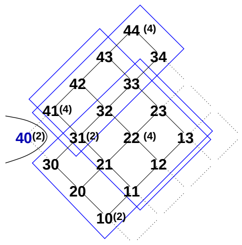

(a) $\{[00, 44]\}$

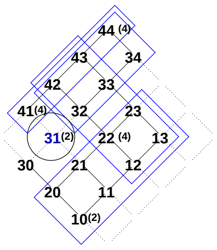

(b) $\{[10, 44], [01, 44]\}$


(c) $\{[10, 44], [04, 44], [01, 42]\}$


(d) $\{[30, 44], [10, 14], [10, 43], [01, 42]\}$


(e) $\{[30, 44], [10, 43]\}$


(f) $\{[31, 44], [10, 33], [11, 43]\}$


(g) $\{[41, 44], [32, 44], [10, 23], [12, 43]\}$

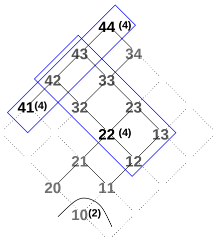
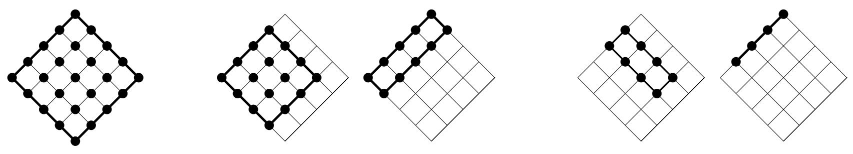

(h) $\{[41, 44], [12, 43]\}$

Figure 8. Maximal intervals after removing elements with label 1 (removing of (a) 00, (b) 03, (c) 24, and (d) 14) followed by minimum cover computation, and after removing elements with label 2 (removing of (e) 40, (f) 31, and (g) 10) followed by minimum cover computation, for the example of Figure 8.

gorithms deal with cases in which some elements are known to be or not to be in the kernel, while nothing is known about the others. In the binary case, the algorithm corresponds to incompletely specified Boolean function minimization. Since these algorithms demand high computational cost, the use of smart data structures and good heuristics need to be considered. A simple example was given to illustrate the use of these algorithms as learning (or generalization) algorithms.

Further issues to be investigated include efficient implementation of the



$\mathbf{B}_0 = \mathbf{B}_1 = \{[00, 44]\}$ $\mathbf{B}_2 = \{[10, 43], [30, 44]\}$ $\mathbf{B}_3 = \mathbf{B}_4 = \{[41, 44], [12, 43]\}$

Figure 9. Resulting basis for the example depicted in Figure 8.

algorithm for the gray-scale case, conversion of an arbitrary representation of the kernel of image operators (such as decision trees, neural networks, support vector machines) to the basis representation, and efficient representation of the basis (aiming fast computation).

Acknowledgments

N. S. T. Hirata thanks FAPESP (Grant 04/11586-7) and CNPq (Grant 312482/2006-0). R. Hirata Jr. and J. Barrera thank CNPq.

References

[1] G. J. F. Banon and J. Barrera, *Minimal Representations for Translation-Invariant Set Mappings by Mathematical Morphology*, SIAM J. Applied Mathematics **51** (1991), no. 6, 1782-1798.

[2] ———, *Decomposition of Mappings between Complete Lattices by Mathematical Morphology, Part I. General Lattices*, Signal Processing **30** (1993), 299-327.

[3] J. Barrera, R. Terada, R. Hirata Jr, and N. S. T. Hirata, *Automatic Programming of Morphological Machines by PAC Learning*, Fundamenta Informaticae **41** (2000), no. 1-2, 229-258.

[4] R. K. Brayton, G. D. Hachtel, C. T. McMullen, and A. L. Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*, Kluwer Academic Publishers, 1984.

[5] H. J. A. M. Heijmans, *Morphological image operators*, Academic, Boston, 1994.

[6] F. J. Hill and G. R. Peterson, *Introduction to Switching Theory and Logical Design*, 3rd, John Wiley, 1981.

[7] N. S. T. Hirata, *Document Processing via Trained Morphological Operators*, To appear in Proceedings of ICDAR 2007, 2007.

[8] N. S. T. Hirata, J. Barrera, R. Terada, and E. R. Dougherty, *The Incremental Splitting of Intervals Algorithm for the Design of Binary Image Operators*, 6th International Symposium on Mathematical Morphology (H. Talbot and R. Beare, ed.), 2002, Proceedings of the 6th International Symposium: ISMM 2002, pp. 219-228.

[9] P. Maragos, *A Unified Theory of Translation-invariant Systems with Applications to Morphological Analysis and Coding of Images*, School of Elect. Eng. - Georgia Inst. Tech., 1985.

[10] ———, *A Representation Theory for Morphological Image and Signal Processing*, IEEE Transaction on Pattern Analysis and Machine Intelligence **11** (1989), no. 6, 586-599.

[11] G. Matheron, *Random Sets and Integral Geometry*, John Wiley, 1975.

Division of mappings between complete lattices

CHRISTER O. KISELMAN

Uppsala University, Sweden
kiselman@math.uu.se

Abstract The importance for image processing of a good theory for morphological operators in complete lattices is now well understood. In this paper we introduce inverses and quotients of mappings between complete lattices which are analogous to inverses and quotients of positive numbers. These concepts are then used to create a convenient formalism for dilations and erosions as well as for cleistomorphisms (closure operators) and anoiktomorphisms (kernel operators).

Keywords: complete lattice, generalized inverse of a mapping, division of mappings, epigraph, hypograph, dilation, erosion, ethmomorphism, morphological filter, cleistomorphism, closure operator, anoiktomorphism, kernel operator, residuated mapping, Galois connection.

1. Introduction

Lattice theory is a mature mathematical theory thanks to the pioneering work by Garrett Birkhoff, Øystein Ore and others in the first half of the twentieth century. A standard reference is still Birkhoff's book (1995) [1], first published in 1940. The importance for image processing of a good theory for morphological operators in complete lattices is now well understood. See for instance the books by Matheron (1975) [9], Serra (1982, 1988) [13], [14] and Heijmans (1994) [6], and the articles by Heijmans and Ronse (1990) [7], Ronse (1990) [11], Ronse and Heijmans (1991) [12], and Serra (2006) [15].

In this paper we shall introduce inverses and quotients of mappings between complete lattices which are analogous to inverses $1/y$ and quotients x/y of positive numbers. These concepts are then used to create a convenient formalism for a unified treatment of dilations $\delta\colon L \to M$ and erosions $\varepsilon\colon L \to M$ as well as of cleistomorphisms (closure operators) $\kappa\colon L \to L$ and anoiktomorphisms (kernel operators) $\alpha\colon L \to L$. The theory for inverses in Section 3 generalizes the theory of Galois connections, which is equivalent to residuation theory.

To define an inverse of a general mapping seems to be a hopeless task. However, if the mapping is between preordered sets, there is some hope of constructing mappings that can serve in certain contexts just like inverses do.

URL of this article: http://urlib.net/dpi.inpe.br/ismm@80/2007/03.01.11.25

2. Definitions

Definition 1. A *preorder* in a set X is a binary relation which is reflexive (for all $x \in X$, $x \leqslant x$) and transitive (for all $x, y, z \in X$, $x \leqslant y$ and $y \leqslant z$ imply $x \leqslant z$). An *order* is a preorder which is antisymmetric (for all $x, y \in X$, $x \leqslant y$ and $y \leqslant x$ imply $x = y$). □

To any preorder $\leqslant$ we introduce an equivalence relation $x \sim y$ defined as $x \leqslant y$ and $y \leqslant x$. If $\leqslant$ is an order, this equivalence relation is just equality. If we have two preorders, we say that $\leqslant_1$ is *stronger than* or *finer than* $\leqslant_2$ if for all x and y, $x \leqslant_1 y$ implies $x \leqslant_2 y$ We also say that $\leqslant_2$ is *weaker than* or *coarser than* $\leqslant_1$. The finest preorder is the *discrete preorder*, defined as equality; the coarsest preorder is the *chaotic preorder* given by $x \leqslant y$ for all x and y.

Definition 2. A *complete lattice* is an ordered set such that any family $(x_j)_{j \in J}$ of elements possesses a smallest majorant and a largest minorant. We denote them by $\bigvee_{j \in J} x_j$ and $\bigwedge_{j \in J} x_j$, respectively. □

Definition 3. If $f\colon X \to Y$ is a mapping of a set into another, we define its *graph* as the set

$$\operatorname{graph} f = \{(x, y) \in X \times Y; y = f(x)\}.$$

If Y is preordered, we define also its *epigraph* and its *hypograph* as

$$\operatorname{epi} f = \{(x, y) \in X \times Y; f(x) \leqslant y\}, \qquad \operatorname{hypo} f = \{(x, y) \in X \times Y; y \leqslant f(x)\}.$$

□

If X and Y are given, any mapping $X \to Y$ is determined by its graph, and, if Y is a complete lattice, also by its epigraph and by its hypograph. It is often convenient to express properties of mappings in terms of their epigraphs or hypographs.

Definition 4. If two preordered sets X and Y and a mapping $f\colon X \to Y$ are given, we shall say that f is *increasing* if

$$\text{for all } x, x' \in X,\ x \leqslant_X x' \ \Rightarrow\ f(x) \leqslant_Y f(x'),$$

and that f is *coincreasing* if

$$\text{for all } x, x' \in X,\ f(x) \leqslant_Y f(x') \ \Rightarrow\ x \leqslant_X x'.$$

□

Blyth and Janowitz (1972:6) [3] and Blyth (2005:5) [2] call an increasing mapping *isotone*. The term *coincreasing* appears in my lecture notes (2002:12) [8].

To emphasize the symmetry between the two notions, we define, given any mapping $f\colon X \to Y$, a preorder $\leqslant_f$ in X by the requirement that

$x \leqslant_f x'$ if and only if $f(x) \leqslant_Y f(x')$. Then f is increasing if and only if $\leqslant_X$ is finer than $\leqslant_f$, and f is coincreasing if and only if $\leqslant_f$ is finer than $\leqslant_X$.

A comparison with topology is in order here. If $f\colon X \to Y$ is a mapping of a topological space X into a topological space Y with topologies τ_X and τ_Y, we can define a new topology τ_f in X as the family of all sets $\{x \in X; f(x) \in V\}$, $V \in \tau_Y$. Then f is continuous if and only if τ_X is finer than τ_f.

Definition 5. A mapping $f\colon L \to M$ of a complete lattice L into a complete lattice M is said to be a *dilation* if $f\left(\bigvee_{j\in J} x_j\right) = \bigvee_{j\in J} f(x_j)$ for all families $(x_j)_{j\in J}$ of elements in L. The mapping is said to be an *erosion* if $f\left(\bigwedge_{j\in J} x_j\right) = \bigwedge_{j\in J} f(x_j)$ for all families $(x_j)_{j\in J}$. □

Singer (1997:172) [16] calls a mapping $f\colon L \to M$ a *duality* if $f\left(\bigwedge_{j\in J} x_j\right) = \bigvee_{j\in J} f(x_j)$ for all families $(x_j)_{j\in J}$ of elements in L. Thus a duality induces a dilation $L^{\mathrm{op}} \to M$ and an erosion $L \to M^{\mathrm{op}}$ if we change the order in L or M to the opposite order; the study of dualities in the sense of Singer is equivalent to that of dilations or erosions.

Definition 6. A mapping $f\colon X \to X$ of a preordered set X into itself is said to be an *ethmomorphism* if it is increasing and idempotent. If in addition it is extensive, i.e., $f(x) \geqslant x$ for all $x \in X$, then it is said to be a *cleistomorphism*; if it is antiextensive, i.e., $f(x) \leqslant x$ for all $x \in X$, then it is called an *anoiktomorphism*.[1] □

For the notions just defined many terms have been used. Other terms for ethmomorphism are *morphological filter* (Serra 1988:104 [14]), *projection operator* and *projection* (Gierz et al. 2003:26 [5]). For cleistomorphism other terms include *closure mapping* (Blyth and Janowitz 1972:9 [3]), *closing* (Matheron 1975:187 [9]; Serra 1982:56 [13]), *hull operator* (Singer 1997:8 [16]), *closure operator* (Gierz et al. 2003:26 [5]). For anoiktomorphism there are several other terms: *dual closure mapping* (Blyth and Janowitz 1972:9 [3]), *opening* (Matheron 1975:187 [9]; Serra 1982:56 [13]), *kernel operator* (Gierz et al. 2003:26 [5]).

3. Inverses of mappings

In general a mapping $g\colon X \to Y$ between sets does not have an inverse. If g is injective, we may define a left inverse $u\colon Y \to X$, thus with $u \circ g = \mathsf{id}_X$. If g is surjective, we may define a right inverse $v\colon Y \to X$, thus with $g \circ v = \mathsf{id}_Y$. We then need to define $v(y)$ as an element of $\{x; g(x) = y\}$. In the general

[1]Cf. the noun *ēthmós* 'strainer' and the adjectives *kleistós* 'closed' and *anoiktós* 'open' in Classical Greek. I am grateful to Ebbe Vilborg for help with these words.

situation this has to be done using the axiom of choice. In a complete lattice, however, it could be interesting to define $v(y)$ as the supremum or infimum of all x such that $g(x) = y$, even though this supremum or infimum need not belong to the set. However, for various purposes it is convenient to take instead the infimum of all x such that $g(x) \geqslant y$ or the supremum of all x such that $g(x) \leqslant y$. This yields better monotonicity properties. (The case $g(x) = y$ is covered if we let the preorder in Y be the discrete preorder.)

Definition 7. Let L be a complete lattice, Y a preordered set, and $g\colon L \to Y$ any mapping. We then define the *upper inverse* $g^{[-1]}\colon Y \to L$ and the *lower inverse* $g_{[-1]}\colon Y \to L$ as the mappings

$$g^{[-1]}(y) = \bigwedge_{x\in L} (x; g(x) \geqslant_Y y) = \bigwedge_{x\in L} (x; (x,y) \in \operatorname{hypo} g), \qquad y \in Y; \tag{1}$$

$$g_{[-1]}(y) = \bigvee_{x\in L} (x; g(x) \leqslant_Y y) = \bigvee_{x\in L} (x; (x,y) \in \operatorname{epi} g), \qquad y \in Y. \tag{2}$$

□

As a first observation, let us note that these inverses are always increasing. If Y possesses a smallest element $\mathbf{0}_Y$, then $g^{[-1]}(\mathbf{0}_Y) = \mathbf{0}_L$. Similarly, if there is a largest element $\mathbf{1}_Y$, then $g_{[-1]}(\mathbf{1}_Y) = \mathbf{1}_L$. If Y has the chaotic preorder, then both inverses are constant, $g^{[-1]} = \mathbf{0}_L$ and $g_{[-1]} = \mathbf{1}_L$ identically.

We note that we always have

$$\left(\operatorname{epi} g^{[-1]}\right)^{-1} \supset \operatorname{hypo} g \text{ and } \left(\operatorname{hypo} g_{[-1]}\right)^{-1} \supset \operatorname{epi} g. \tag{3}$$

Here $S^{-1} = \{(y,x) \in Y \times L; (x,y) \in S\}$ for any subset S of $L \times Y$. In general these inclusions are strict as we shall see below.

Note that we do not require in (2) that the set of all x such that $g(x) \leqslant_Y y$ shall have a largest element. In other words, the supremum in (2) is not necessarily a maximum.

The special situation when the supremum in (2) is a maximum, in other words when $g(g_{[-1]}(y)) \leqslant_Y y$ for all y, has been studied for a long time, and from various aspects. Let us mention a few examples.

1. When the supremum in (2) is a maximum, the pair $(g, g_{[-1]})$ is said to be a *Galois connection* (Gierz et al. 2003:22) [5], a concept which goes back to Évariste Galois' work on the automorphism groups of a field. Ore (1944:495) [10] called a variant of the pair of mappings $(g, g_{[-1]})$ a *Galois connexion.*

2. One also says in this special case that g is *residuated* and that $g_{[-1]}$ is its *residual* (Blyth and Janowitz 1972:11 [3]; Blyth 2005:7 [2]). If the infimum in (1) is a minimum, g is said to be *dually residuated* and $g^{[-1]}$ is called its *dual residual*; the pair $(g^{[-1]}, g)$ is a Galois connection between Y and L. Residuation theory goes back at least to a paper by Ward and Dilworth (1939) [17]. In an ordered groupoid one fixes an element c and

assumes that the set of all x such that $cx \leqslant y$ has a largest element, which is denoted by $y : c$ (we consider for simplicity only the commutative case). We see that this is $g_{[-1]}$ if $g\colon L \to L$ is the mapping $g(x) = cx$. Thus $cx \leqslant y$ if and only if $x \leqslant y : c$.

3. The pair $(g, g_{[-1]})$ is also said to be an *adjunction* (Gierz et al. 2003:22 [5]) in this special case. This aspect probably originates in logic, and is important in image processing.

4. Finally, there is duality in convexity theory. The Fenchel transformation (Fenchel 1949 [4]) of a function $\varphi\colon \mathbf{R}^n \to [-\infty, +\infty]$ is defined as

$$\tilde{\varphi}(\xi) = \sup_{x \in \mathbf{R}^n} \big(\xi \cdot x - \varphi(x)\big), \qquad \xi \in \mathbf{R}^n,$$

and satisfies

$$\tilde{\varphi} \leqslant f \iff \tilde{f} \leqslant \varphi.$$

After a change of order on one of the sides it satisfies (3) with equality, which means that we have a Galois connection (see condition (C) in Theorem 2). It is also the case that

$$\big(\inf_{j \in J} \varphi\big)\tilde{} = \sup_{j \in J} \tilde{\varphi},$$

so that we have a duality in the sense of Singer; i.e., after a change of the order relation we have a dilation or erosion (see condition (A) in Theorem 2). Singer (1997) [16] studies several other dualities in convexity theory.

The results of the present section generalize residuation theory, equivalently the theory of Galois connections, to a more general situation, a situation which appears even in very simple examples as we shall see now.

It seems that this generalization of residuation theory has not been considered in the contexts of the branches of mathematics mentioned under 1, 2, and 3 above. However, Singer (1997:176) [16] defines the *dual* $M \to L$ of a duality $L \to M$, which, after a change of order in L, is the lower inverse defined here. He notes the inclusion corresponding to the second inclusion in (3) and proves that it is an equality when g is a dilation.

Example 1. Take $Y = L$ in Definition 7, fix an element c of L, and define a mapping $g\colon L \to L$ by $g(x) = x \vee c$, $x \in L$. In this case, the supremum in (2) is a maximum if $y \geqslant c$ but only then. Thus g is not residuated unless $c = \mathbf{0}$. But it is easy to determine its lower inverse: $g_{[-1]}(y) = y$ if $y \geqslant c$ and $g_{[-1]}(y) = \mathbf{0}$ otherwise. We have

$$\operatorname{epi} g = \{(x, y) \in L^2; y \geqslant x \vee c\},$$

while

$$\big(\operatorname{hypo} g_{[-1]}\big)^{-1} = \operatorname{epi} g \cup \{(\mathbf{0}, y) \in L^2; y \not\geqslant c\},$$

so that

$$\big(\operatorname{hypo} g_{[-1]}\big)^{-1} \smallsetminus \operatorname{epi} g = \{(\mathbf{0}, y) \in L^2; y \not\geqslant c\} \neq \varnothing \text{ if } c \neq \mathbf{0}.$$

For the upper inverse, we can only say that $g^{[-1]} = \mathbf{0}$ if $y \leqslant c$ and that $g^{[-1]}(y) \leqslant y$ for $y \not\leqslant c$. Both equality and strict inequality can occur here as we shall see. □

Example 2. We let g be as in Example 1 and assume in addition that L is totally ordered. We have already determined $g_{[-1]}$ in Example 1, and we know that $g^{[-1]}(y) = \mathbf{0}$ for $y \leqslant c$. In the case of total order, we have $g^{[-1]}(y) = y$ for all $y > c$. In the notation which Singer (1997:335) [16] uses for $L = [-\infty, +\infty]$, we can write $g^{[-1]}(y) = y \top c$, $y \in L$. Thus $g_{[-1]}$ and $g^{[-1]}$ are equal except for $y = c$, where we get $g^{[-1]}(c) = \mathbf{0} \leqslant c = g_{[-1]}(c)$. Moreover we have

$$\left(\operatorname{epi} g^{[-1]}\right)^{-1} = \operatorname{hypo} g = \{(x, y) \in L^2; y \leqslant x \vee c\},$$

which, in view of Corollary 1 means that $g^{[-1]}$ is dually residuated with dual residual g, or that $(g^{[-1]}, g)$ is a Galois connection. □

Example 3. Let now L be $[0, 1]^2$, the Cartesian product of two intervals. The lower inverse is already known from Example 1. The upper inverse is

$$g^{[-1]}(y) = \begin{cases} \mathbf{0}, & y \leqslant c; \\ (0, y_2), & y_1 \leqslant c_1,\ y_2 > c_2; \\ (y_1, 0), & y_1 > c_1,\ y_2 \leqslant c_2; \\ y, & y_1 > c_1, y_2 > c_2. \end{cases}$$

Thus strict inequality in $g^{[-1]}(y) \leqslant y$ can occur. We have $\left(\operatorname{epi}(g^{[-1]})\right)^{-1} = \operatorname{hypo} g$. □

Example 4. Let now L be $\{0, 1\}^2$ with the coordinatewise order, and let g be as in Example 1. We choose $c = (1, 0)$ and denote $(0, 1)$ by a so that L consists of the four element $\mathbf{0} = (0, 0)$, $a = (0, 1)$, $c = (1, 0)$, and $\mathbf{1} = (1, 1)$. From Example 1 we know that $g_{[-1]}(y) = y$ if $y \geqslant c$ and $g(y) = \mathbf{0}$ otherwise. Thus

$$g_{[-1]}(y) = \begin{cases} \mathbf{0}, & y = \mathbf{0}; \\ \mathbf{0}, & y = a; \\ c, & y = c; \\ \mathbf{1}, & y = \mathbf{1}. \end{cases}$$

We find that

$$\left(\operatorname{hypo} g_{[-1]}\right)^{-1} \smallsetminus \operatorname{epi} g = \{(\mathbf{0}, \mathbf{0}), (\mathbf{0}, a)\} \neq \varnothing.$$

Thus g is not residuated.

We also find that

$$g^{[-1]}(y) = \begin{cases} \mathbf{0}, & y = \mathbf{0}; \\ a, & y = a; \\ \mathbf{0}, & y = c; \\ a, & y = \mathbf{1}. \end{cases}$$

The infimum is in all cases a minimum, meaning that $g^{[-1]}$ is dually residuated, in other words, $(g^{[-1]}, g)$ is a Galois connection. We have

$$\big(\operatorname{epi} g^{[-1]}\big)^{-1} = \operatorname{hypo} g = L^2 \smallsetminus \{(\mathbf{0}, a), (\mathbf{0}, \mathbf{1}), (c, a), (c, \mathbf{1})\}.$$

□

If, given a mapping $g\colon L \to Y$, we can find a mapping $u\colon Y \to L$ such that $\operatorname{epi} u = (\operatorname{hypo} g)^{-1}$ we would be content to have a kind of inverse to g. However, usually the best we can do is to study mappings with $\operatorname{epi} u \supset (\operatorname{hypo} g)^{-1}$ or $\operatorname{epi} v \subset (\operatorname{hypo} g)^{-1}$. This we shall do in the following proposition, which shows that the upper and lower inverses are solutions to certain extremal problems.

Proposition 1. *Let L be a complete lattice, Y a preordered set, and let $g\colon L \to Y$, $u, v\colon Y \to L$ be mappings. If $\operatorname{epi} u \supset (\operatorname{hypo} g)^{-1} \supset \operatorname{epi} v$, then $u \leqslant g^{[-1]} \leqslant v$ and*

$$\operatorname{epi} u \supset \operatorname{epi} g^{[-1]} \supset (\operatorname{hypo} g)^{-1} \supset \operatorname{epi} v.$$

Hence $g^{[-1]}$ is the largest mapping u such that $\operatorname{epi} u$ contains $(\operatorname{hypo} g)^{-1}$. Similarly, if $\operatorname{hypo} u \subset (\operatorname{epi} g)^{-1} \subset \operatorname{hypo} v$, then $u \leqslant g_{[-1]} \leqslant v$ and

$$\operatorname{hypo} u \subset (\operatorname{epi} g)^{-1} \subset \operatorname{hypo} g_{[-1]} \subset \operatorname{hypo} v.$$

Hence $g_{[-1]}$ is the smallest mapping v which satisfies $\operatorname{hypo} v \supset (\operatorname{epi} g)^{-1}$.

Corollary 1. *With g, u and v given as in the proposition, assume that $(\operatorname{epi} u)^{-1} = \operatorname{hypo} g$. Then $u = g^{[-1]}$. Similarly, if $(\operatorname{hypo} v)^{-1} = \operatorname{epi} g$, then $v = g_{[-1]}$. If also Y is a complete lattice, then $\operatorname{epi} u = (\operatorname{hypo} g)^{-1}$ implies that $u_{[-1]} = g$ in addition to $u = g^{[-1]}$. Similarly, $\operatorname{hypo} v = (\operatorname{epi} g)^{-1}$ implies $v^{[-1]} = g$ in addition to $v = g_{[-1]}$.*

The corollary singles out the special case of adjunctions between L and Y among all pairs $\big(g, g_{[-1]}\big)$ and adjunctions between Y and L among all pairs $\big(g^{[-1]}, g\big)$.

An ideal inverse u would satisfy $u \circ g = \mathsf{id}_L$, $g \circ u = \mathsf{id}_Y$, and the inverse of u would be g. It is therefore natural to compare $g^{[-1]} \circ g$ and $g_{[-1]} \circ g$ with id_L; $g \circ g^{[-1]}$ and $g \circ g_{[-1]}$ with id_Y; and $\big(g_{[-1]}\big)^{[-1]}$ and $\big(g^{[-1]}\big)_{[-1]}$ with g. This is what we shall do now.

Left inverses

We shall now investigate to what extent $g^{[-1]}$ and $g_{[-1]}$ can serve as left inverses to g.

Proposition 2. *Suppose that L is a complete lattice and Y a preordered set. Then for all mappings $g\colon L \to Y$ one has $g^{[-1]} \circ g \leqslant \mathsf{id}_L \leqslant g_{[-1]} \circ g$.*

The following three conditions are equivalent:
(α) g is coincreasing;
(β) $g^{[-1]} \circ g = \mathsf{id}_L$;
(γ) $g_{[-1]} \circ g = \mathsf{id}_L$.

Corollary 2. *Let L be a complete lattice and Y a preordered set. Then $g^{[-1]}(y) \leqslant g_{[-1]}(y)$ for all $y \in \operatorname{im} g$, and also for all y majorizing or minorizing* $\operatorname{im} g$. *In particular, $g^{[-1]} \leqslant g_{[-1]}$ if g is surjective.*

Proposition 3. *If u, v are increasing mappings such that $u \circ g \leqslant \mathsf{id}_L \leqslant v \circ g$, then $u \leqslant g^{[-1]}$ and $v \geqslant g_{[-1]}$. Hence, in view of Proposition 2, $g^{[-1]}$ is the largest increasing mapping u such that $u \circ g \leqslant \mathsf{id}_L$, and $g_{[-1]}$ is the smallest increasing mapping v such that $v \circ g \geqslant \mathsf{id}_L$.*

Theorem 1. *Let L be a complete lattice and Y a preordered set. Then the following six conditions are equivalent.*
(a) *g is coincreasing;*
(b) *$g^{[-1]} \circ g \geqslant \mathsf{id}_L$;*
(c) *$g^{[-1]} \circ g = \mathsf{id}_L$;*
(d) *$g_{[-1]} \circ g \leqslant \mathsf{id}_L$;*
(e) *$g_{[-1]} \circ g = \mathsf{id}_L$;*
(f) *$g_{[-1]} \leqslant g^{[-1]}$.*

Right inverses

Next we compose $g_{[-1]}$ with g in the other order: we shall see to what extent the inverses we have constructed can serve as right inverses. This will lead to a characterization of dilations—and, by duality, of erosions.

Theorem 2. *If L and M are complete lattices and $g\colon L \to M$ is any mapping, then the following five properties are equivalent.*
(A) *g is a dilation;*
(B) *$\big(\operatorname{hypo}(g_{[-1]})\big)^{-1} \subset \operatorname{epi} g$;*
(C) *$\big(\operatorname{hypo}(g_{[-1]})\big)^{-1} = \operatorname{epi} g$;*
(D) *g is increasing and $\big(\operatorname{graph}(g_{[-1]})\big)^{-1} \subset \operatorname{epi} g$;*
(E) *g is increasing and $g \circ g_{[-1]} \leqslant \mathsf{id}_M$.*

This theorem characterizes the special case when the supremum in (2) is a maximum (Property (E)); equivalently, it characterizes the special case of residuated mappings or Galois connections (Property (C)).

By duality we get a similar characterization of erosions; equivalently of the case when the infimum defining the upper inverse is a minimum.

Singer (1997:178, Proposition 5.3) [16] proves that (A) and (E) are equivalent (expressed for dualities).

Corollary 3. *If L and M are complete lattices and $g\colon L \to M$ and $u\colon M \to L$ are two mappings such that* $\operatorname{epi} g = (\operatorname{hypo} u)^{-1}$, *then u is a dilation and g is an erosion, and $g_{[-1]} = u$, $u^{[-1]} = g$.*

Inverses of inverses

Theorem 3. *If L and M are complete lattices and $g\colon L \to M$ is any mapping, then quite generally $\big(g_{[-1]}\big)^{[-1]} \leqslant g \leqslant \big(g^{[-1]}\big)_{[-1]}$. Equality holds at the first place if and only if g is a dilation; at the second place if and only if g is an erosion.*

Theorem 4. *If L and M are complete lattices and $\delta\colon L \to M$ is a dilation, then $\delta_{[-1]}\colon M \to L$ is an erosion. Similarly, if $\varepsilon\colon L \to M$ is an erosion, then $\varepsilon^{[-1]}$ is a dilation.*

Corollary 4. *For any dilation $\delta\colon L \to M$ we have $\delta \circ \delta_{[-1]} \circ \delta = \delta$ and $\delta_{[-1]} \circ \delta \circ \delta_{[-1]} = \delta_{[-1]}$. In particular, $\delta_{[-1]} \circ \delta$ and $\delta \circ \delta_{[-1]}$ are idempotent and therefore ethmomorphisms. The first is a cleistomorphism in L, the second an anoiktomorphism in M. Dually $\varepsilon \circ \varepsilon^{[-1]} \circ \varepsilon = \varepsilon$ and $\varepsilon^{[-1]} \circ \varepsilon \circ \varepsilon^{[-1]} = \varepsilon^{[-1]}$ for any erosion $\varepsilon\colon L \to M$. Also $\varepsilon^{[-1]} \circ \varepsilon$ and $\varepsilon \circ \varepsilon^{[-1]}$ are idempotent; the first an anoiktomorphism, the second a cleistomorphism.*

4. Division of mappings

We shall now generalize the definitions of upper and lower inverses.

Definition 8. Let a set X, a complete lattice M, and a preordered set Y, as well as two mappings $f\colon X \to M$ and $g\colon X \to Y$ be given. We define two mappings $f/^{\star}g, f/_{\star}\, g\colon Y \to M$ by

$$(f/^{\star}g)(y) = \bigwedge_{x\in X} (f(x); g(x) \geqslant_Y y), \qquad y \in Y,$$

$$(f/_{\star}\, g)(y) = \bigvee_{x\in X} (f(x); g(x) \leqslant_Y y), \qquad y \in Y.$$

We shall call them the *upper quotient* and the *lower quotient* of f and g. □

We shall often assume that X, M and Y are all complete lattices, but this is not necessary for the definitions to make sense.

The quotients $f/^{\star}g$ and $f/_{\star}\, g$ increase when f increases and they decrease when g increases—just as with division of positive numbers:

If $f_1 \leqslant_M f_2$ and $g_1 \geqslant_Y g_2$, then $f_1/^{\star}g_1 \leqslant_M f_2/^{\star}g_2$ and $f_1/_{\star}\, g_1 \leqslant_M f_2/_{\star}\, g_2$.

The mappings $f/^{\star}g$ and $f/_{\star}\, g$ are always increasing. If $g(x) \geqslant_Y y$, then $f(x) \geqslant_M (f/^{\star}g)(y)$; if $g(x) \leqslant_Y y$, then $f(x) \leqslant_M (f/_{\star}\, g)(y)$. In particular, if $g(x) = y$, then $(f/^{\star}g)(y) \leqslant_M f(x) \leqslant_M (f/_{\star}\, g)(y)$.

If we specialize the definitions to the situation when $X = M$ and $f = \mathsf{id}_X$, then $f /^{\star} g = \mathsf{id}_X /^{\star} g = g^{[-1]}$ and $f /_{\star} g = \mathsf{id}_X /_{\star} g = g_{[-1]}$; cf. Definition 7.

We note another special case:

Proposition 4. *For all mappings $f\colon X \to M$ we have*

$$f /_{\star} f \leqslant \mathsf{id}_M \leqslant f /^{\star} f$$

and

$$(f /_{\star} f) \circ f = f = (f /^{\star} f) \circ f. \tag{4}$$

Proposition 5. *Let X be an arbitrary subset of a complete lattice M, let $Y = M$, and g the inclusion mapping $X \to M$. Then $f /^{\star} g = f^{\diamond}$ and $f /_{\star} g = f_{\diamond}$, where $f^{\diamond}$ is the largest increasing mapping $h\colon M \to M$ such that $h\big|_X$ minorizes f, i.e.,*

$$f^{\diamond}(y) = \sup_h \big(h(y); h \text{ is increasing and } h(x) \leqslant f(x) \text{ for all } x \in X\big);$$

and $f_{\diamond}$ is the smallest increasing mapping k such that $k\big|_X$ majorizes f, i.e.,

$$f_{\diamond}(y) = \inf_k \big(k(y); k \text{ is increasing and } k(x) \geqslant f(x) \text{ for all } x \in X\big).$$

If f itself is increasing, they are in fact extensions of f.

The definitions of $f^{\diamond}$ and $f_{\diamond}$ are taken from Matheron (1975:187) [9] and are generalized here to any complete lattice.

If we specialize further, letting also f be the inclusion mapping $X \to M$, we obtain

$$(f /_{\star} g)(y) = (f /_{\star} f)(y) = f_{\diamond}(y) = \bigvee_{x \in X} (x; x \leqslant y) = y^{\circ} \in M,$$

where the last equality defines y°. It is easy to verify that $y \mapsto y^{\circ}$ is an anoiktomorphism. A well-known situation is described in the following example.

Example 5. Let M be the complete lattice $[-\infty, +\infty]^E$ of functions on a vector space E with values in the extended reals, let F be a vector subspace of its algebraic dual $E^{\star}$ (the space of all linear forms on E), and let X be the set of all affine functions with linear part in F, i.e., functions of the form $\alpha(x) = \xi(x) + c$ for some linear form $\xi \in F$ and some real constant c. A function f such that $f^{\circ} = f$ is called *X-convex* by Singer (1997:10) [16].

We see that a function on E is X-convex in the sense of Singer if and only if it is equal to the supremum of all its affine minorants belonging to X.

We may ask for a characterization of the X-convex functions. A generalization of Fenchel's theorem to this setting gives the answer: this happens if and only if the function possesses three properties:

(a) it is convex in the usual sense;
(b) it is lower semicontinuous for the topology $\sigma(E, F)$ on E generated by the linear forms in F; and
(c) it does not take the value $-\infty$ except when it is equal to the constant $-\infty$. □

Proposition 6. *If $f\colon X \to M$ is increasing and $g\colon X \to Y$ is coincreasing, then $f /_{\star} g \leqslant f /^{\star} g$.*

The upper quotient $f /^{\star} g$ is the optimal solution to an inequality:

Proposition 7. *For all mappings $f\colon X \to M$ and $g\colon X \to Y$ we have*

$$(f /^{\star} g) \circ g \leqslant f \leqslant (f /_{\star} g) \circ g,$$

with equality if f is increasing and g is coincreasing. From this we deduce that $(f /^{\star} g)(y) \leqslant (f /_{\star} g)(y)$ for all $y \in \operatorname{im} g$ as well as for all majorants and minorants of $\operatorname{im} g$. In particular, $f /^{\star} g \leqslant f /_{\star} g$ if g is surjective.

Conversely, if $u, v\colon Y \to M$ are two increasing functions such that $u \circ g \leqslant f \leqslant v \circ g$, then $u \leqslant f /^{\star} g$ and $v \geqslant f /_{\star} g$. Thus $f /^{\star} g$ is the largest increasing function u such that $u \circ g \leqslant f$, and $f /_{\star} g$ is the smallest increasing function v such that $f \leqslant v \circ g$.

In the special case $X = Y$ and $g = \mathsf{id}_X$ we obtain

$$f /^{\star} \mathsf{id}_X \leqslant f \leqslant f /_{\star} \mathsf{id}_X,$$

where $f /^{\star} \mathsf{id}_X$ is the largest increasing minorant of f and $f_{\star} /\mathsf{id}_X$ is the smallest increasing majorant of f; when f itself is increasing we therefore get equality.

We next compare the quotient $f /^{\star} g$ and the composition $f \circ g^{[-1]}$ (think of $x/y = x \cdot y^{-1}$ for positive numbers):

Proposition 8. *For every increasing mapping $f\colon X \to M$ and every mapping $g\colon X \to Y$ we have $f /^{\star} g \geqslant f \circ g^{[-1]}$ with equality if f is an erosion, and $f /_{\star} g \leqslant f \circ g_{[-1]}$ with equality if f is a dilation. If g is coincreasing, then $f /_{\star} g \leqslant f \circ g_{[-1]} \leqslant f \circ g^{[-1]} \leqslant f /^{\star} g$.*

Proposition 9. *If P is a preordered set and $h\colon M \to P$ is increasing, we have $h \circ (f /^{\star} g) \leqslant (h \circ f) /^{\star} g$ with equality if h is an erosion. Similarly $h \circ (f /_{\star} g) \geqslant (h \circ f) /_{\star} g$ with equality if h is a dilation. A special case is $h \circ (f /^{\star} \mathsf{id}_X) \leqslant (h \circ f) /^{\star} \mathsf{id}_X$ (take $X = Y$ and $g = \mathsf{id}_X$). Another special case is Proposition 8 (take $X = M$ and $f = \mathsf{id}_X$).*

Cleistomorphisms and anoiktomorphisms

Theorem 5. *Let $f\colon X \to M$ be any mapping from a set X into a complete lattice M. Then $\alpha = f /_{\star} f\colon M \to M$ is an anoiktomorphism. Conversely, any anoiktomorphism in M is of this form for some mapping $f\colon X \to M$ with $X = M$. By duality we get analogous statements for the upper quotient and cleistomorphisms.*

5. Conclusion

We have introduced the notions of upper and lower inverses and upper and lower quotients for mappings between complete lattices. Their most basic properties have been investigated, in particular how the inverses can serve as left and right inverses to a given mapping. Important morphological operators can be systematically treated in the calculus created. In particular, anoiktomorphisms are always lower quotients of the form $f/_{\star} f$, and cleistomorphisms are always upper quotients of the form $f/^{\star}f$.

References

[1] G. Birkhoff, *Lattice Theory*, American Mathematical Society Colloquium Publications, vol. 25. Third edition, eighth printing, Providence, RI, 1995, xiii + 420 pp.

[2] T. S. Blyth, *Lattices and Ordered Algebraic Structures*, Springer, 2005, X + 303 pp.

[3] T. S. Blyth and M. F. Janowitz, *Residuation Theory*, Pergamon Press., Oxford et al., 1972, ix + 379 pp.

[4] W. Fenchel, On conjugate convex functions, *Canadian J. Math.* **1** (1949), 73–77.

[5] G. Gierz, K. H. Hofmann, K. Keimel, J. D. Lawson, M. Mislove, and D. S. Scott, *Continuous lattices and domains*, Cambridge University Press, Cambridge, 2003, xxxvi + 591 pp.

[6] H. J. A. M. Heijmans, *Morphological image operators*, Academic Press, Boston, 1994.

[7] H. J. A. M. Heijmans and Christian Ronse, The algebraic basis of mathematical morphology. I. Dilations and erosions, *Comput. Vision Graphics Image Process.* **50** (1990), 245–295.

[8] C. O. Kiselman, *Digital Geometry and Mathematical Morphology*, Uppsala University. Lecture Notes, 85 pp., available at `www.math.uu/~kiselman`, Uppsala, 2002.

[9] G. Matheron, *Random Sets and Integral Geometry*, Wiley & Sons, New York et al., 1975, xxv + 261 pp.

[10] Ø. Ore, Galois connexions, *Trans. Amer. Math. Soc.* **55** (1944), 493–513.

[11] C. Ronse, Why mathematical morphology needs complete lattices, *Signal Processing* **21** (1990), 129–154.

[12] C. Ronse and H. J. A. M. Heijmans, The algebraic basis of mathematical morphology. II. Openings and closings, *Comput. Vision Graphics Image Process.: Image Understanding* **54** (1991), no. 1, 74–97.

[13] J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, London, 1982.

[14] ———, Introduction to morphological filters. In: J. Serra (ed.), *Image Analysis and Mathematical Morphology, Vol. 2, Theoretical Advances,* pp. 101–114. Academic Press, London et al., 1988, xvi + 411 pp.

[15] ———, A lattice approach to image segmentation, *J. Math. Imaging Vision* **24** (2006), no. 1, 83–130.

[16] I. Singer, *Abstract Convex Analysis*, John Wiley & Sons., New York et al., 1997, xix + 491 pp.

[17] M. Ward and R. P. Dilworth, Residuated lattices, *Trans. Amer. Math. Soc.* **45** (1939), no. 3, 335–354.

General approach for fuzzy mathematical morphology

Antony T. Popov

Faculty of Mathematics and Informatics, University of Sofia "St. Kliment Ohridski", Bulgaria
atpopov@fmi.uni-sofia.bg

Abstract This work shows how a generalized approach for constructing dilation-erosion adjunctions on fuzzy sets can be defined using appropriate chosen complete lattice. Some applications in the field of computation with uncertainties are given, more precisely in the interval arithmetic and the calculations with fuzzy numbers. Applications to image segmentation such as geodesic operations and reconstruction are given as well. Also we discuss how intuitionistic fuzzy sets can be used as structuring elements for fuzzy morphological operations, especially in fuzzy hit-or-miss transform. The aim is to find objects with close to given shape and size on digital images.

Keywords: complete lattice, fuzzy sets, interval and arithmetic operation, fuzzy arithmetic operation, hit-or-miss transform, T-invariant operation, geodesic operation.

1. Introduction

There are several approaches for fuzzifying mathematical morphology, see for instance [1]. In our work we step on the framework of Deng and Heijmans (see for details [3]) based on conjunctors-implicators adjoint fuzzy logical operators. We generalize this definition presenting a universal framework and we define naturally fuzzy geodesic morphological operations. Also, this model is applied to fuzzy arithmetic, built by analog to the interval arithmetic [8] which makes possible the definition of inner addition and multiplication of fuzzy numbers. On the other hand, in the pioneering works of Serra [11] and Heijmans [5] it is demonstrated that the hit-or-miss transform plays an essential role in shape analysis. So, here we define general fuzzy hit-or-miss transform for grey-scale image segmentation and we show how it is related to the theory of intuitionistic fuzzy sets (IFS).

In this work we use the same notions and notations about complete lattices and the morphological operations on them as in [5]. For instance, let $\mathcal{L}$ be a complete lattice with a supremum generating family l, and let T be an Abelian group of automorphisms of $\mathcal{L}$ acting transitively over l. The elements of T are denoted by τ_x, namely for any $x \in l$, $\tau_x(o) = x$, where o

G. Banon, et al. (eds),
Mathematical Morphology and its Applications to Signal and Image Processing, 39–48.
©2007 *MCT/INPE. Printed in Brazil.*
URL of this article: http://urlib.net/dpi.inpe.br/ismm@80/2007/03.19.13.52

is a fixed element of l interpreted as an origin. Then also, we can consider a symmetry in $\mathcal{L}$ as $\check{A} = \bigvee_{a \in l(A)} \tau_a^{-1}(o)$. Evidently $\check{a} = \tau_a^{-1}(o) = \tau_{\check{a}}(o)$ for any $a \in l$. If A is an arbitrary element of the lattice $\mathcal{L}$ let us denote by $l(A) = \{a \in l \mid a \leq A\}$ the supremum-generating set of A. Following [5] we define the operations

$$\delta_A = \bigvee_{a \in l(A)} \tau_a \tag{1}$$

and

$$\varepsilon_A = \bigwedge_{a \in l(A)} \tau_a^{-1} = \bigwedge_{a \in l(A)} \tau_{\check{a}}, \tag{2}$$

which form an adjunction. δ_A and ε_A are T-invariant operators called *dilation and erosion by the structuring element A.* Remind that a pair of operators (ε, δ) between two lattices, $\varepsilon : \mathcal{M} \mapsto \mathcal{L}$ and $\delta : \mathcal{L} \mapsto \mathcal{M}$ is called an *adjunction* if for every two elements $X \in \mathcal{L}$ and $Y \in \mathcal{M}$ it follows that

$$\delta(X) \leq Y \iff X \leq \varepsilon(Y).$$

In [5] it is proved that if (ε, δ) is an adjunction then ε is erosion and δ is dilation. In other hand, every dilation $\delta : \mathcal{L} \mapsto \mathcal{M}$ has a unique adjoint erosion $\varepsilon : \mathcal{M} \mapsto \mathcal{L}$, and vice-versa.

2. Fuzzy sets and fuzzy morphological operations

Consider the set E called the universal set. A fuzzy subset A of the universal set E can be considered as a function $\mu_A : E \mapsto [0, 1]$, called the membership function of A. $\mu_A(x)$ is called the degree of membership of the point x to the set A. The ordinary subsets of E, sometimes called 'crisp sets', can be considered as a particular case of a fuzzy set with membership function taking only the values 0 and 1.

Let $0 < \alpha \leq 1$. An α-cut of the set X (denoted by $[X]_\alpha$) is the set of points x, for which $\mu_X(x) \geq \alpha$.

The usual set-theoretical operations can be defined naturally on fuzzy sets: Union and intersection of a collection of fuzzy sets is defined as supremum, resp. infimum of their membership functions. Also, we say that $A \subseteq B$ if $\mu_A(x) \leq \mu_B(x)$ for all $x \in E$. The complement of A is the set A^c with membership function $\mu_{A^c}(x) = 1 - \mu_A(x)$ for all $x \in E$. If the universal set E is linear, like the d-dimensional Euclidean vector space $\mathbb{R}^d$ or the space of integer vectors with length d, then any geometrical transformation arising from a point mapping can be generalised from sets to fuzzy sets by taking the formula of this transformation for graphs of numerical functions, i.e., for any transformation ψ like scaling, translation, rotation etc. we have that $\psi(\mu_A(x)) = \mu_A(\psi^{-1}(x))$. Therefore we can transform fuzzy sets by transforming their $\alpha-$cuts like ordinary sets. Further on, for simplicity, we shall write simply $A(x)$ instead of $\mu_A(x)$.

Say that the function $c(x, y) : [0, 1] \times [0, 1] \mapsto [0, 1]$ is conjunctor conjunctor if c is increasing in the both arguments, $c(0, 1) = c(1, 0) = 0$, and $c(1, 1) = 1$. We say that a conjunctor is a *t-norm* if it is commutative, i.e. $c(x, y) = c(y, x)$, associative $c(c(x, y), z) = c(x, c(y, z))$ and $c(x, 1) = x$ for every number $x \in [0, 1]$, see for instance [1,6].

Say that the function $i(x, y) : [0, 1] \times [0, 1] \mapsto [0, 1]$ is implicator implicator if i is increasing in y and decreasing in x, $i(0, 0) = i(1, 1) = 1$, and $i(1, 0) = 0$.

In [3] a number of adjoint conjunctor-implicator pairs are proposed. Here we give examples of two of them:

$$\begin{aligned} c(b, y) &= \min(b, y), \\ i(b, x) &= \begin{cases} x & x < b, \\ 1 & x \geq b. \end{cases} \end{aligned}$$

$$\begin{aligned} c(b, y) &= \max(0, b + y - 1), \\ i(b, x) &= \min(1, x - b + 1). \end{aligned}$$

The first pair is known as operations of Gödel-Brouwer, while the second pair is suggested by Lukasiewicz.

Also, a widely used conjunctor is $c(b, y) = by$, see [6]. Its adjoint implicator is

$$i(b, x) = \begin{cases} \min\left(1, \frac{x}{b}\right) & b \neq 0, \\ 1 & b = 0. \end{cases}$$

2.1 General definition of fuzzy morphology

There are different ways to define fuzzy morphological operations. An immediate paradigm for defining fuzzy morphological operators is to lift each binary operator to a grey-scale operator by fuzzifying its primitive composing operations. However thus we rarely obtain erosion-dilation adjunctions, which leads to non-idempotent openings and closings. Therefore we use the idea from [3], saying that having an adjoint conjunctor-implicator pair, we can define a fuzzy erosion-dilation adjunction.

So let us consider the universal set E and a class of fuzzy sets $\{A_y, \mid y \in E\}$. Then for any fuzzy subset X of the universal set E, let us define fuzzy dilation and erosion as follows:

$$\delta(X)(x) = \bigvee_{y \in E} c(A_x(y), X(y)), \tag{3}$$

$$\varepsilon(X)(x) = \bigwedge_{y \in E} i(A_y(x), X(y)). \tag{4}$$

To prove that this pair of operations is an adjunction, let us consider the case $\delta(X) \subseteq Z$ in fuzzy sense, which means that for every $x, y \in E$ $c(A_x(y), X(y)) \leq Z(x)$. Then $X(y) \leq i(A_x(y), Z(x))$ for all $x, y \in E$.

Since $\varepsilon(Z)(y) = \bigwedge_{x\in E} i(A_x(y), Z(x))$, then we have that $X \subseteq \varepsilon(Z)$, which ends the proof.

3. How to define T-invariant and geodesic fuzzy morphological operations?

Let us consider a universal set E. Let also there exists an Abelian group of automorphisms T in $\mathcal{P}(E)$ such that T acts transitevely on the supremum-generating family $l = \{\{e\} | e \in E\}$ as defined previously. In this case, for shortness we shall say that T *acts transitively on* E. Then having an arbitrary fuzzy subset B from E, we can define a family of fuzzy sets in $\{A^B_y \,|\, y \in E\}$ such as $A^B_y(x) = B(\tau_y^{-1}(x))$. Recall that for any $\tau \in T$ there exists $y \in E$ such that $\tau = \tau_y$, and for any fuzzy subset M we have that $(\tau(M))(x) = M(\tau^{-1}(x))$. Then having in mind Equations 3 and 4 we can define a fuzzy adjunction by the *structuring element* B by:

$$\delta_B(X)(x) = \bigvee_{y\in E} c(A^B_x(y), X(y)), \tag{5}$$

$$\varepsilon_B(X)(x) = \bigwedge_{y\in E} i(A^B_y(x), X(y)). \tag{6}$$

We show that that the upper operations are T-invariant. To prove this statement, following [5], it is sufficient to show that every such erosion commutes with an arbitrary automorphism τ_b for any $b \in E$. Evidently

$$\varepsilon_B(\tau_b(X))(x) = \bigwedge_{y\in E} i(B(\tau_y^{-1}(x)), X(\tau_b^{-1}(y))).$$

suppose that $\tau_b^{-1}(y) = z$, which means that $\tau_y = \tau_z\, \tau_b$. Then

$$\varepsilon_B(\tau_b(X))(x) = \bigwedge_{z\in E} i(B(\tau_z^{-1}(\tau_b^{-1}(x)), X(z)) = \varepsilon_B(X)(\tau_b^{-1}(x)),$$

which simply means that $\varepsilon_B(\tau_b(X)) = \tau_b(\varepsilon_B(X))$, which ends the proof.

Now consider that in E we have a continuous commutative operation $* : E \times E \mapsto E$. Then let us define $\tau_b(x) = b * x$. In the case of Gödel-Brouwer conjunctor-implicator pair the respective dilation has the form

$$(\delta_B(X))(x) \;=\; \bigvee_{y*z=x} \min(X(y), B(z)).$$

Following the *extension principle* (see [6]) for the definition of the operation $X * B$ between the fuzzy sets X and B over the universal set E, it is evident that in this case

$$\delta_B(X) = \delta_X(B) = X * B.$$

In [6] it is proved that

$$[X * B]_\alpha = [X]_\alpha * [B]_\alpha = \{z \in E |\, z = a * b, a \in [X]_\alpha, b \in [B]_\alpha\}.$$

Following [10], let us say that the points $x, y \in E$ are connected in the fuzzy set A if there exists a path Γ from x to y such that

$$\inf_{z\in\Gamma} A(z) \geq \min(A(x), A(y)).$$

Let now M be a fuzzy subset of the universal set E, which is a numerical metric space. Then if x and y are two points from E which are connected in M, we can define the following *geodesic distance between x and y* [2]:

$$d_M(x, y) = \frac{\operatorname{len}(x, y)}{C_M(x, y)}, \tag{7}$$

where $\operatorname{len}(x, y)$ is the length of the shortest continuous path between x and y due to the metric in E, and

$$C_M(x, y) = \sup_{\Gamma} \inf\{M(z) |\, z \in \Gamma\}.$$

Here Γ denotes an arbitrary path between x and y in E. Since we are working with almost connected objects, while a point has membership 0 when it is from the background (i.e., it does not belong to any object on the scene), we may suppose that C_M is always positive. The quantity $d_M(x, y)$ satisfies all properties of a metrics except the triangle inequality, so it is not a real distance. However, if M is a crisp set, then it is equal to the classical geodesic distance. Now we can define a fuzzy geodesic ball

$$[B_M(y, r)](x) = \begin{cases} 1 & \text{if } d_M(x, y) \leq r, \\ 0 & \text{otherwise.} \end{cases}$$

Having in mind the expressions (3)-(4) and (5)-(6) we can define a fuzzy geodesic adjunction $(\mathcal{E}_M^r, \Delta_M^r)$ as:

$$\Delta_M^r(X)(x) = \bigvee_{y\in E} c[(B_M(x, r))(y), X(y)],$$
$$\mathcal{E}_M^r(X)(x) = \bigwedge_{y\in E} i[(B_M(y, r))(x), X(y)].$$

Therefore we can define fuzzy geodesic reconstruction and idempotent fuzzy geodesic openings and closings as in the binary case descibed in [13]. An example of the usage of this operation is given in [9].

4. Interval computations and computations with fuzzy numbers

Interval computations are computations using intervals with the aim to guarantee the result in particular in the presence of data uncertainties and rounding errors. Since the α-cuts of the fuzzy numbers are closed intervals, then the interval calculus is essential part of the computations with fuzzy numbers. A fuzzy number is a fuzzy subset of $\mathbb{R}$, i.e. it represents a generalization of a real number r. Any fuzzy number A satisfies the following conditions ([6]):

- $A(x) = 1$ for exactly one x;
- the support of A is bounded;
- the α-cuts of A are closed intervals.

In [8] it is shown that there exists a close relation between interval and morphological operations. Having in mind this relation and our general definition of fuzzy morphological operations, we can express the known arithmetic operations between fuzzy numbers through morphological ones and thus we can define inner operations. As shown in [8], the outer and inner interval operations are related to binary dilations and erosions as follows:

$$A + B = A \oplus B = \delta_A(B) = \delta_B(A),$$
$$A +^- B = A \ominus (-B) \cup B \ominus (-A) = \varepsilon_{-B}(A) \cup \varepsilon_{-A}(B).$$

Now let denote by $F(\mathbb{R})$ the set of fuzzy numbers. Then we can define following operations on them using the *extension principle* [6]:

$$(A + B)(x) = \bigvee_{z+y=x} \min(A(y), B(z));$$

$$(A \times B)(x) = \bigvee_{z.y=x} \min(A(y), B(z));$$

$$(A - B)(x) = \bigvee_{y-z=x} \min(A(y), B(z)) = (A + (-B))(x);$$

$$\frac{A}{B}(x) = \bigvee_{zx=y} \min(A(y), B(z)) = \left(A \times \frac{1}{B}\right)(x).$$

Note that every real number r could be considered as fuzzy number with membership function, which is zero on the whole real line, except in r where it takes value 1.

The sum, the difference and the product of fuzzy numbers are also fuzzy numbers. The division is always possible, however the result is a fuzzy number only when $0 \notin \mathrm{supp}(B)$. In general, the quotient is a fuzzy quantity over the real line which support may not be bounded. Also, if A and B are fuzzy numbers then $[A+B]_\alpha = [A]_\alpha + [B]_\alpha$ and $[A\times B]_\alpha = [A]_\alpha \times [B]_\alpha$. Now consider the group of automorhisms $\tau_b(x)$ in $\mathbb{R}$ and the fuzzy operations on $F(\mathbb{R})$ defined by Gödel-Brouwer conjunctor-implicator pair:

$$(\delta_B(A))(x) = \bigvee_{y*z=x} \min(A(y), B(z)),$$

$$(\varepsilon_B(A))(x) = \inf_{y\in\mathbb{R}} \left(h\left(A(y) - B(\tau_x^{-1}(y)\right)(1 - A(y)) + A(y)\right),$$

where $h(x) = 1$ when $x \geq 0$ and is zero otherwise.

Now it is clear that if $\tau_b(x) = x + b$ and $* = +$ then

$$(\delta_B(A)) = A + B.$$

We can also define an *inner addition* operation by

$$A +^- B = \varepsilon_{-B}(A) \cup \varepsilon_{-A}(B).$$

If $\tau_b(x) = xb$ for $b \neq 0$ and $y * z = yz$ then

$$(\delta_B(A)) = A \times B.$$

In this case an inner multiplication exists as well:

$$A \times^- B = \varepsilon_{\frac{1}{B}}(A) \cup \varepsilon_{\frac{1}{A}}(B).$$

Note that in this definition we can work with fuzzy numbers which do not contain 0 in their support. It is not difficult to show directly that $A +^- B \subseteq A + B$ and $A \times^- B \subseteq A \times B$.

5. Fuzzy hit- or- miss transform and intuitionistic fuzzy sets

Remind that an intuitionistic fuzzy subset A from the universal set E is characterised by two functions: the degree of membership $\mu_A(x)$ and the degree of nonmembership $\nu_A(x)$. As described in [4], for every point $x \in E$ we have that $\mu_A(x) + \nu_A(x) \leq 1$. Then one can define intersection ot two intuitionistic sets by taking a t-norm Δ of their membership functions for the resulting membership function, and taking the associated s-norm ∇ of their nonmembership functions for the resulting nonmembership functions. Remind that the associated s-norm is defined by $x\nabla y = 1-((1-x)\Delta(1-y))$. For the union of two intuitionistic sets we take s-norm for the membership part and the respective t-norm for the nonmembership part.

It is natural to lift to the fuzzy framework the hit-or-miss morphological operator

$$\tilde{\pi}_{A,B}(X) = \varepsilon_A(X)\Delta\, \varepsilon_B(X^c).$$

A key difference with the binary case is that, since A and B are fuzzy sets, we do not assume that $A \cap B = \emptyset$. Note that here we can use any T-invariant fuzzy operations. Further considerations are done in case of usual translation invariance. The experiments indicate that, in the case of noisy images, it is preferable for the structuring elements to be slightly fuzzy, which means that the values of their membership functions have to be close to one in their support. Note that traditional hit-or-miss operation with crisp templates would only mark the objects in the original word but would not in the noisy realizations. Unlike its classical counterpart, the fuzzy hit-or-miss operation always marked the desired objects. To express clearly in a table how a intuitionistic fuzzy set with a finite domain looks like, let us denote by a/b the membership and nonmembership degree of a given element. This means that if we consider an intuitionistic structuring element, then for any pixel x we use the notation $\mu_A(x)/\nu_A(x)$ to show the respective values in the table. If both values are zero, we simply write 0 at the appropriate place in the table. Note, that the origin is located always at the central element of the table. An example of the usage of such "combine" structuring element for a noisy image is given on Figure 1. The element is described on Table 1. The task is to find a 'c'-shaped pattern with a given size on a grey-scale image. The marked 'c'-shape arround the handle (pointed by an arrow) has been detected with degree of truth 0.54. Similar examples for using such patterns, used to detect given characters in a text, can be found in [7].

Table 1. The hit-or-mis structuring element for finding a shape like the letter 'c'.

0	0	0	0.6/0.2	0.6/0.2	0.6/0.2	0	0	0
0	0	0	0.6/0	1/0	0.6/0	0	0	0
0	0	0	0.6/0	0.6/0	0.6/0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0.6/0.2	0.6/0	0.6/0	0	0	0/0.8	0/0.8	0/0.8	0/0.8
0.6/0.2	1/0	0.6/0	0	0/1	0/1	0/1	0/1	0/1
0.6/0.2	0.6/0	0.6/0	0	0	0/0.8	0/0.8	0/0.8	0/0.8
0	0	0	0	0	0	1/0	0	0
0	0	0	0.8/0	0.8/0	0.8/0	0	0	0
0	0	0	0.8/0	1/0	0.8/0	0	0	0
0	0	0	0.6/0.2	0.6/0.2	0.6/0.2	0	0	0

Interesting applications of intuitionistic models in image processing are given in [14]. Further we are going to experiment the fuzzy hit-or-miss transform by intuitionistic elements for finding skeleta by thinning and pseudoconvex hulls by thickenning and to make experiments with intuitionistic elements based on other fuzzy adjoint operations.

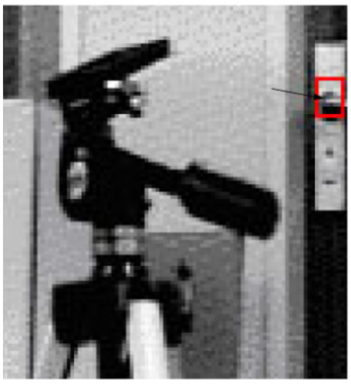


Figure 1. Finding a 'c'-shaped pattern.

Acknowledgments

The author would like to thank Prof. Ed Dougherty and Prof. Henk Heijmans for their help. This work was partially supported by National Ministry of Science and Education under contracts VU-TN-202/2006 "Methods and algorithhms for analysis of combined biometric information" and VU-MI-204/2006 "Intelligent sensor systems for security improvement".

References

[1] I. Bloch and H. Maître, *Fuzzy mathematical morphologies: a comparative study*, Pattern Recognition **28**, no. 9, 1341–1387.

[2] I. Bloch, *Geodesic balls in a fuzzy set and fuzzy geodesic mathematical morphology*, Pattern Recognition **33**, no. 6, 897–905.

[3] T.-Q. Deng and H. J. A. M. Heijmans, *Grey-scale morphology based on fuzzy logic, vol. PNA-R0012*, CWI, Amsterdam, 2000.

[4] G. Gargov and K. Atanassov, *On the intuitionistic fuzzy logic operations*, Notes on IFS **1** (1995), no. 1, 1–4.

[5] H. J. A. M. Heijmans, *Morphological image operators*, Academic, Boston, 1994.

[6] H. T. Nguyen and E. A. Walker, *A first course in fuzzy logic (2nd edition)*, CRC Press, Boca Raton FL, 2000.

[7] E. R. Dougherty and A. T. Popov, *Fuzzy mathematical morphology based on fuzzy inclusion* (M. Nachtegael and E. E. Kerre, eds.), Physica-Verlag, Berlin, 2000, Studies in Fuzziness and Soft Computing, Vol. 52 : Fuzzy Techniques in Image Processing, Chapter 3, pp. 76–100.

[8] A. T. Popov, *Numerical and approximation tools based on mathematical morphology*, International Symposium on Mathematical Morphology, 4 (Amsterdam, June 3–5, 1998) (H. J. A. M. Heijmans and J. B. T. M. Roerdink, eds.), Kluwer Academic Publishers, Dordrecht, 1998, Mathematical Morphology and its Applications to Image and Signal Processing, pp. 3–10.

[9] ———, *General definition of fuzzy mathematical morphology Operations. Image and non-image applications* (M. Nachtegael, D. Van der Weken, E. E. Kerre, and W. Philips, eds.), Springer, Berlin, 2007, Studies in Fuzziness and Soft Computing, Vol. 210 : Soft Computing in Image Processing - Recent Advances, Chapter 13, pp. 353–381.

[10] A. Rosenfeld, *The fuzzy geometry of image subset*, Pattern Recognition Letters **2** (1984), no. 2, 311–317.

[11] J. Serra, *Image analysis and mathematical morphology*, Academic, London, 1982.

[12] ———, *Mathematical morphology for complete lattices* (J. Serra, ed.), Academic, London, 1988, Image analysis and mathematical morphology, vol.2.

[13] P. Soille, *Morphological image analysis (2nd edition)*, Springer, Berlin, 2002.

[14] I. K. Vlachos and G. D. Sergiadis, *Intuitionistic fuzzy image processing* (M. Nachtegael, D. Van der Weken, E. E. Kerre, and W. VPhilips, eds.), Springer, Berlin, 2007, Studies in Fuzziness and Soft Computing, Vol. 210 : Soft Computing in Image Processing - Recent Advances, Chapter 14, pp. 383–414.

Self-dual morphology on tree semilattices and applications

Alla Vichik*, [1], Renato Keshet†, [2] and David Malah[1]

[1] *Electrical Engineering Department, Technion–Israel Institute of Techlonogy, Haifa, Israel*
`malah@ee.technion.ac.il`

[2] *Hewlett Packard Laboratories–Israel, Technion City, Haifa, Israel*
`renato.keshet@hp.com`

Abstract We present a new tree-based framework for producing self-dual morphological operators. For any given tree representation of images, one can associate a complete inf-semilattice (CISL) in the corresponding tree-representation domain, where the operators can then be derived. We also present a particular case of this general framework, involving a new tree representation, the Extrema-Watershed Tree (EWT). The operators obtained by using the EWT in the above framework behave like classical morphological operators, but in addition are self-dual. Some application examples are provided: pre-processing for OCR and dust & scratch removal algorithms, and image denoising.

Keywords: complete inf-semilattices, self-dual operators, tree representation of images.

1. Introduction

One of the main approaches for producing *self-dual*[1] morphological operators is by means of a tree representation. For instance, Salembier and Garrido proposed a *Binary Partition Tree* for hierarchical segmentation in [12, 15]. A *tree of shapes* was proposed by Monasse and Guichard [10, 11] (see also [1, 2]). These tree representations are usually used for performing *connected* filtering operations on an image; however, they do not yield non-connected operators, such as erosions, dilations or openings by a structuring element.

In [7] (see also [8]) a new complete inf-semilattice (CISL), called the shape-tree semilattice, was introduced. This semilattice provides non-connected morphological operations, based on the above-mentioned "tree of

*`alla.vichik@gmail.com`

†Renato Keshet is also an adjunct lecturer at the EE dep., Technion–Israel Institute of Techlonogy.

[1]An operator ψ is self-dual when $\psi(-f) = -\psi(f)$ for all input f.

shapes". As a consequence, self-dual erosions and openings were obtained. Similar operators had been developed earlier on the so-called Reference Semilattices (introduced in [6], and further studied by Heijmans and Keshet in [4]); however, they require a reference image, which somewhat limits the usage of these operators. The self-dual operators in the shape-tree semilattice provide erosions and openings without the need for a reference image.

In this paper, we present a general framework for tree-based morphological image processing, which generalizes the shape-tree operators. This framework yields a set of new morphological operators (erosion, dilation, opening, etc.), for each given tree representation of images. The heart of the proposed approach is a novel complete inf-semilattice of tree representations of images. Because many of the properties of the tree are inherited by the corresponding operators, the choice of the tree representation is of high importance. We focus mostly on self-dual trees, which represent dark and bright elements equally.

A particular case of the proposed framework is also presented, based on a novel tree representation, the Extrema-Watershed Tree (EWT). Following the general framework, we derive self-dual morphological operators from the EWT. Examples of applications discussed here are pre-processing for OCR (Optical Character Recognition) algorithms, de-noising of images, and pre-processing for dust and scratch removal.

2. Theoretical background

2.1 Complete inf-semilattices

A complete inf-semilattice (CISL) is a partially-ordered set $\mathcal{S}$, where the infimum operation ($\wedge$) is always well-defined (but the supremum $\vee$ is not necessarily so). The theory of mathematical morphology on complete semilattices was introduced in [5], and is an almost-straightforward extension of the traditional morphology on complete lattices. It mathematically supports intuitive observations, such as the fact that erosions are naturally extended from complete lattices to CISLs, whereas dilations are not universally well-defined on CISLs.

On the other hand, some results may not be necessarily intuitive. The main ones are as follows: (a) it is always possible to associate an opening γ to a given erosion ε by means of $\gamma(x) = \bigwedge\{y \mid \varepsilon(y) = \varepsilon(x)\}$, (b) even though the adjoint dilation δ is not universally well-defined, it is always well defined for elements on the image of $\mathcal{S}$ by ε, and (c) $\gamma = \delta\varepsilon$.

2.2 Rooted trees and their corresponding CISL

This section reviews basic graph theory notions (given in [3, chapter 1]), including the natural partial ordering on rooted trees, which provide them with a CISL structure.

A *graph* is a pair of sets $G = (V, E)$ satisfying $E \subseteq [V]^2$. A *path* is a non-empty graph $P = (V, E)$ of the form: $V = \{x_0, x_1, ..., x_k\}$, $E = \{x_0x_1, x_1x_2, ..., x_{k-1}x_k\}$, where the x_i are all distinct. A cycle is a path where $k \geq 2$ and $(x_0, x_k) \in E$. A graph not containing any cycles, is called a *forest*. A connected forest is called a tree (thus, a forest is a graph whose components are trees).

Sometimes it is convenient to consider one vertex of a tree as special; such a vertex is then called the *root* of this tree. A tree with a fixed root is a rooted tree. Choosing a root r in a tree t imposes the following partial ordering on V(t): $x \preceq_t y \iff x \in rty$, where rty is the unique path in t that connects y to the root. Note that $(V, \preceq_t)$ is a CISL, where r is the least element, and the maximal elements are the leafs of t. The infimum between vertices is the nearest common ancestor vertex.

We say that a tree t_1 is smaller than another tree t_2 if $t_1 \subseteq t_2$.

3. Tree semilattices

This section presents the proposed general framework for tree-based morphological image processing (introduced in [16]). This framework enables the definition of new morphological operators that are based on tree representations. The proposed image processing scheme is shown is Figure 1.

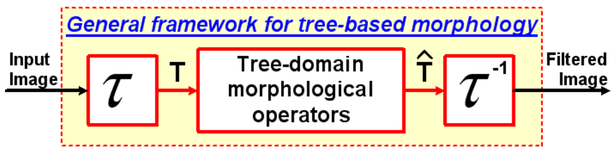

General framework for tree-based morphology
Input Image
$\mathcal{T}$
T
Tree-domain morphological operators
$\hat{T}$
$\mathcal{T}^{-1}$
Filtered Image

Figure 1. Tree-based morphology.

3.1 CISL of tree representations

The heart of the proposed approach is a novel complete inf-semilattice of tree representations of images. Let L be an arbitrary set of "labels", and let $t = (V, E)$ be a rooted tree, with root r, such that $V \subseteq L$. Therefore t is a tree of labels. Moreover, let $M : \mathbb{E} \mapsto V$ be an image of vertices, mapping each point in $\mathbb{E}$ to a vertex of t. As usual, $\mathbb{E}$ may be an Euclidean space or a discrete rectangular grid within the image area.

Definition 1. (Tree representation) The structure $T = (t, M)$ shall be called a *tree representation*. The set of all tree representations associated with the label set L and with the root r shall be denoted by $\mathcal{T}_r^L$. □

Consider the following relation between tree representations: For all $T_1 = (t_1, M_1)$ and $T_2 = (t_2, M_2)$ in $\mathcal{T}_r^L$,

$$T_1 \leq T_2 \iff \begin{cases} t_1 \subseteq t_2 \text{ and} \\ M_1(x) \preceq_{t_2} M_2(x), \forall x \in \mathbb{E}, \end{cases} \tag{1}$$

where $\subseteq$ is the usual graph inclusion, and $\preceq_{t_2}$ is the partial ordering of vertices within the tree t_2 (see Subsection 2.2).

Proposition 1. *The above tree relation $\leq$ is a partial ordering on $\mathcal{T}_r^L$, and $\left(\mathcal{T}_r^L, \leq\right)$ is a CISL. The least element is $T_0 \triangleq ((\{r\}, \{\}), M_0(x) \equiv r))$.*

The proof is given in [16]. The general format of the corresponding infimum and supremum operators are also derived in [16]. Here, however, we focus on the particular case where all tree presentations involved in an infimum or supremum operation have a common tree associated with them:

Proposition 2. *Let $\{T_i = (t, M_i)\}$ be a collection of tree representations with a common tree t. In this case,*

$$\bigwedge_i T_i = (t, \curlywedge_t \{M_i\}), \tag{2}$$

and

$$\bigvee_i T_i = (t, \curlyvee_t \{M_i\}), \tag{3}$$

where $\curlywedge_t$ and $\curlyvee_t$ are the point-wise infimum and supremum associated to vertex order $\preceq_t$, respectively. Notice that $\curlyvee_t \{M_i\}$ may not always exist.

The situation where the set of tree representations share the same tree is what one encounters when defining flat erosions and dilations on the complete inf-semilattice of tree representations. The flat erosion can be defined as the operator ε given by:

$$\varepsilon_B(T) \triangleq \bigwedge_{b \in B} T_{-b} = \bigwedge_{b \in B} (t, M_{-b}), \tag{4}$$

where B is a structuring element. It is easy to verify that the above operator is indeed an erosion on $\mathcal{T}_r^L$.

Using Proposition 2, one obtains that

$$\varepsilon_B(T) = (t, \curlywedge_t \{M_{-b} | b \in B\}). \tag{5}$$

As reminded in Section 2.1, on a complete inf-semilattice, one can associate to any given erosion ε an opening γ (and, in fact, any morphological operator that is derived from compositions of erosions and openings, such as the internal gradient, dark top-hat transform, and skeletons). Furthermore, the adjoint dilation δ exists, and, even though it is not well defined for all complete inf-semilattice elements, it is always well-defined for elements that are mapped by the erosion, and $\gamma = \delta\varepsilon$.

In the case of the above tree-representation flat erosion, the adjoint dilation is given by:

$$\delta_B(T) = (t, \curlyvee_t \{M_b | b \in B\}) . \tag{6}$$

We also define the tree-representation reconstruction of T from a marker $\overline{T} = (t, \overline{M}) \le T$ as the infinite iteration of the conditional dilation

$$\delta_B(T|\overline{T}) \overset{\triangle}{=} \left(t, \curlyvee_t \left\{M_b \curlywedge_t \overline{M} | b \in B\right\}\right) . \tag{7}$$

Notice that $\delta_B(T|\overline{T})$ is always well defined, since it consists of a supremum of bounded elements.

3.2 Image processing on tree semilattices

Now that morphology on the tree representation domain has been established, we can turn to our ultimate goal, which is to process a given grayscale image f. Let us assume that f is an integer-valued function on $\mathbb{E}$, i.e., $f \in \text{Fun}(\mathbb{E}, \mathbb{Z})$. Moreover, let τ by an operator that transforms f into a pair (T, ℓ) , where $T = (t, M) \in \mathcal{T}_r^L$ is a tree representation, and $\ell : L \mapsto \mathbb{Z}$ is a function that maps labels into graylevels. The tree transformation τ should be invertible, and the inversion be given by: $\tau^{-1}(\ell, M(x)) = \ell\,(M(x))$. We propose the following approach for processing f, using the CISL of tree representations:

1. compute $\tau(f) = (T, \ell)$;
2. perform one or more morphological operations on T to obtain a processed tree representation $\hat{T} = (t, \hat{M})$;
3. transform $(\hat{T}, \ell)$ back into a new image $\hat{f} \in \text{Fun}(\mathbb{E}, \mathbb{Z})$, using:
$$\hat{f}(x) = \tau^{-1}(\ell, \hat{M}(x)) = \ell\left(\hat{M}(x)\right) . \tag{8}$$

If the morphological operation in Step 2 above is the erosion ε_B, then all three steps can be collapsed into the following equation:

$$\hat{f}(x) = \ell\left(\curlywedge_t \{M_{-b}(x) | b \in B\}\right) . \tag{9}$$

Proposition 3. *For any vertex v in V, let $R(v) \overset{\triangle}{=} \{x \in \mathbb{E} | M(x) \succeq_t v\}$ and $\hat{R}(v) \overset{\triangle}{=} \{x \in \mathbb{E} | \hat{M}(x) \succeq_t v\}$, where $\hat{M}$ is again the mapping function after the erosion ε_B. Then, for all v:*

$$\hat{R}(v) = R(v) \ominus B, \tag{10}$$

where $(.) \ominus B$ is the traditional binary erosion by the s.e. B.

Proposition 3 (which is proven in [16]) suggests an alternative algorithm for computing the erosion. For any v, (a) compute $R(v)$, (b) compute $\hat{R}(v) = R(v) \ominus B$, and (c) assign $\ell(v)$ to all points within $\hat{R}(v) \setminus \bigcup_{v \prec v'} \hat{R}(v')$.

3.3 Particular cases and examples

In order for the tree transform to be invertible, τ should be such that it assigns a common label to each flat zone of f. This is because τ^{-1} maps each label to a single graylevel. This suggests that special attention should be paid to the flat zones of f.

One way of addressing the flat zones of a given image is by considering its Regional Adjacency Graph (RAG). The RAG is a graph, where V is the set of all flat zones of the image, and E contains all pairs of flat zones that are adjacent to each other.

A spanning tree is a subgraph of a RAG that should, obviously, be a tree, and have the same vertex set V as the RAG. A spanning tree creates a hierarchy in the RAG, defining father/son relationships between adjacent flat zones.

The proposed morphological scheme is of particular interest when t is a spanning tree of the RAG. In this case, the associated morphological operators do not create new grey/color values.

One particular group of trees are the Max- and Min-Trees [13]. When a tree vertex is always brighter (resp., darker) then its sons, as in the Max-Tree (resp., Min-Tree), the infimum operation always changes the gray level to the local minimum (resp., maximum), which is precisely what the traditional grayscale erosion (resp., dilation) does. In other words, for these trees, the proposed tree approach becomes the traditional grayscale mathematical morphology (resp., its dual version).

More interesting particular cases are the Boundary Topographic Variation (BTV) Tree (see [16]), which is built from the RAG using a minimal topographic distance criterion. Another one is the shape-tree defined in [9] and the resulting semilattice defined in [7, 8]. Both provide self-dual morphological operators, based on some inclusion criterion.

3.4 Image semilattice

What we would really like is the CISL of tree representation (using a tree τ) to induce a CISL in the image domain. That is, we would like, for instance, the composite operation of $\tau^{-1}\varepsilon\tau$ to be an erosion in the image domain. However, that is not guaranteed. In fact, the partial ordering in the tree-representation domain does induce a partial ordering for images, for any τ; however, the infimum operation is not guaranteed to be well defined. This issue is still under study.

4. Extrema-watershed tree

Based on the general framework of Section 3, all that is needed in order to obtain a new set of morphological operators is a given tree representation. In this section, we explore a particular case of the proposed framework, using a novel self-dual tree representation, which we call the *Extrema-Watershed Tree* (EWT). The EWT is a particular case of "Binary Partitioning Tree" [12]; in particular, the proposed representation is built using a particular case of the iterative merging process presented by Salembier, Garrido and Garcia in [14], as follows:

Input all the extrema of a given image (i.e., all regions associated to a local minimum or maximum) into a list, sorted by increasing area[2]. Also, initiate the EWT by setting each flat zone as a leaf vertex. The main loop for the computation of the EWT is as follows: Take the first extremum from the ordered list (the one with smallest area), and merge it with the adjacent neighbor that is the closest one in terms of graylevels. Then, set the merged region as the parent vertex of the above two regions (the extremum and its neighbor) in the EWT. Select the graylevel of the non-extremum region to be the graylevel of the new merged region. Finally, check whether the newly merged region and all its neighbors are extrema, and insert those that are into the sorted list (in their corresponding place, according to the listing order). This loop runs until the list has just one element, which then becomes the EWT root.

Figure 2 illustrates the computation of the EWT. Consider the image in Figure 2(a), which contains two extrema with the same area: v_1 and v_3. The first step of the procedure, shown in Figure 2(b), consists of merging v_1 with v_2, since the difference in graylevel between v_1 and v_2 is smaller than the one between v_3 and v_4. This merger produces a new flat zone – v_5, with the same graylevel as v_2 – which is a new extremum in the image. In the next step, shown in Figure 2(c), the extremum v_3 is merged with v_4 to create v_6. The procedure continues until all extrema (old and new) are merged. Figure 2(d) shows the final EWT.

As described in Section 3, once a tree transform is defined, morphological operations (such as erosion and opening) in the tree-domain can be derived. The new operators typically inherit some of the properties of the tree, such as self-duality, for instance. Figure 3 shows the result of the EWT erosion and EWT opening. Notice that very small features are removed, whereas the larger ones shrunk, in a self-dual manner. The average gray level of the picture does not change; in particular, the picture does not become darker, which is what usually happens after a standard erosion or opening.

[2]If two extrema have the same area, input first the one who has the smallest grayscale distance to its closest neighbor.

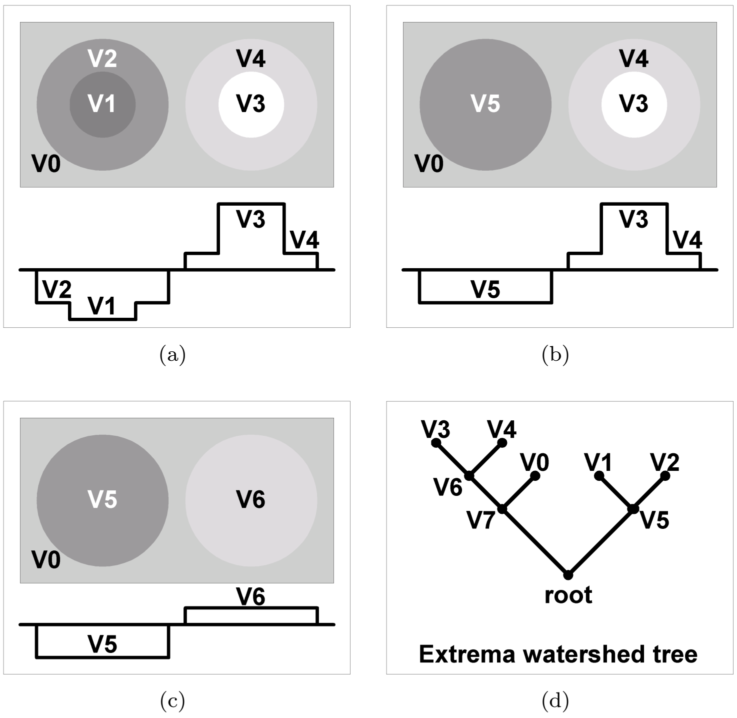


Figure 2. Example of the EWT computation. (a) Input image, (b) first merging step, (c) second merging step, (d) the final EWT.

5. Application examples

The EWT has many potential applications; in this section we list just a few.

One application that requires image simplification is pre-processing for OCR (Optical Character Recognition). We have chosen a specific OCR algorithm, used for recognition of license plate numbers, that was developed in [17]. This algorithm uses a mask for each digit and looks for the best correlation among these masks with an image. The algorithm also outputs a confidence grade, which can be used for comparing algorithms. Any noise that exists in the image degrades the correlation value and interferes with the recognition. Consider the example license plate shown in Figure 4(a), which has been artificially corrupted with blobs of different sizes. Without pre-processing the algorithm fails to read the correct number. Several different algorithms (including linear filtering and traditional grayscale morphology) has been applied to this image. In order to compensate for the lack of duality in classical morphology, we have also compared the EWT with the “quasi-self-dual” Opening-Closing by reconstruction operator. The

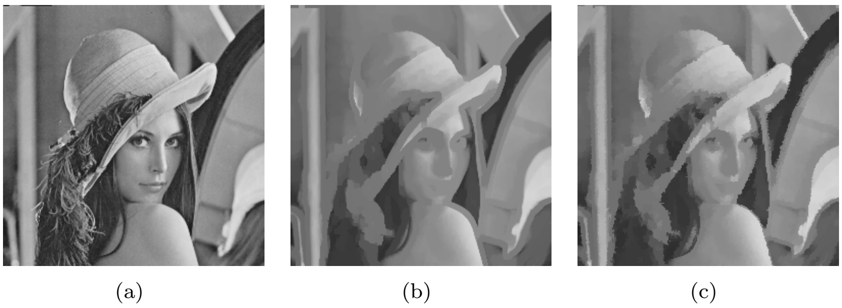


Figure 3. (a) Original image, (b) EWT erosion by square SE 5×5, (c) EWT opening by square SE 5×5.

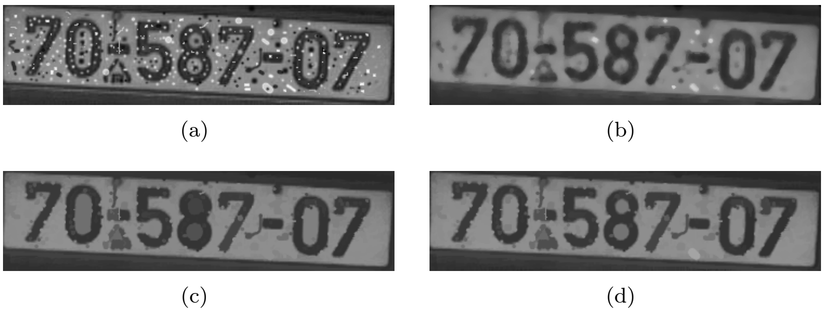


Figure 4. (a) Input image, artificially corrupted, (b) filtered with a median filter, (c) filtered with regular self dual opening by reconstruction, and (d) filtered by the EWT-based opening by reconstruction, using circle SE of radius 4.

only algorithms that cause the algorithm to correctly read the number were the median filter, quasi-self-dual filter and the EWT opening by reconstruction (see Figure 4(b), 4(c) and 4(d), respectively). The confidence grades associated with the EWT pre-processed image were higher than those for the median filter and the quasi-self-dual filter. Further details on this experiment can be found in [16].

Another example uses opening by reconstruction as an initial step for an application that removes dust and scratches from images. The elements filtered by the opening by reconstruction are completely extracted, including their edges. This enables one to extract candidates for dust and scratch removal, without corrupting their shapes. The proposed operation is a EWT top-hat filter. Figure 5 shows an example. Later steps (not considered here) can then make further analysis of the image in order to decide which candidates should be removed. We have compared the proposed approach to linear and median filters. Subjective and objective criteria were used.

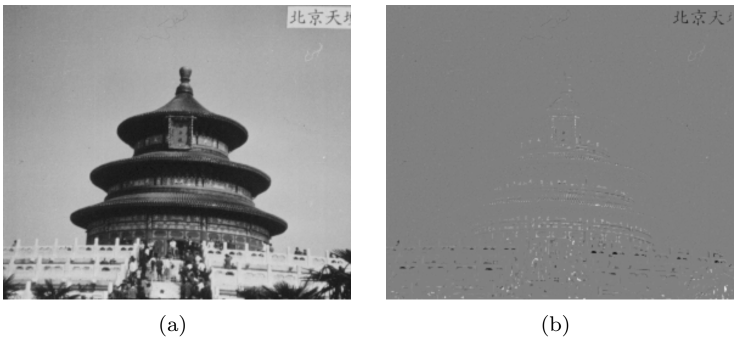


Figure 5. Top hat, using cross SE 3×3, as a pre-processing stage for dust and scratch removal. (a) Original image (b) Top hat by reconstruction based on EWT.

The subjective criterion is the overall corruption of the candidate shapes. The objective criterion is the measured energy of the filtered images. The EWT performed better in both criteria. On one hand, for the relevant structuring elements, the energy of the EWT filtered image was lower than for the linear and median filters. On the other hand, the linear and median filters do not completely extract the artifacts, as can be seen in Figure 6 for the “cross” structuring element.

6. Conclusion

We have presented a general framework for producing new morphological operators that are compatible to given tree representations. Furthermore, a useful particular case is provided, based on a new tree representation, the Extrema Watershed Tree. The resulting morphological erosion and opening operators were applied to a number of application examples, giving better results in comparison to other filtering techniques, including classical morphological filtering. In general, EWT-based filtering performs well in tasks suitable for classical morphological filtering, especially when self-duality is required.

7. Acknowledgments

We would like to thank the reviewers for their thorough remarks, and, in particular, the 2nd reviewer, for the insights regarding the reconstruction operator.

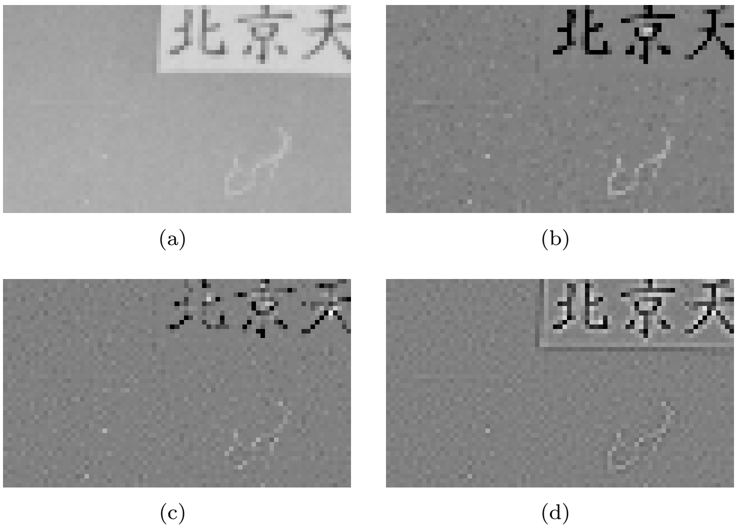


Figure 6. Zoom in. Top hat, using cross SE 3 × 3, as a pre-processing stage for dust and scratch removal. (a) Original image (b) Top hat by reconstruction based on EWT (c) Top hat using median (d) Top hat using an averaging filter.

References

[1] C. Ballester, V. Caselles, and P. Monasse, *The Tree of Shapes of an Image*, ESAIM: Control, Optimization and Calculus of Variations **9** (2003), 1-18.

[2] V. Caselles and P. Monasse, *Grain filters*, Journal of Mathematical Imaging and Vision **17** (November 2002), no. 3, 249-270.

[3] R. Diestel, *Graph Theory*, Electronic Edition, Springer-Verlag New York, 2000.

[4] H. J. A. M. Heijmans and R. Keshet, *Inf-semilattice approach to self-dual morphology*, Journal of Mathematical Imaging and Vision **17** (July 2002), 55-80.

[5] R. Keshet, *Extension of Morphological Operations to Complete Semilattices and Its Applications to Image and Video Processing*, Mathematical Morphology and its Applications to Image and Signal Processing (Proc. of ISMM'98) (June 1998), 35-42.

[6] ———, *Mathematical Morphology on Complete Semilattices and its Applications to Image Processing*, Fundamenta Informaticæ **41** (January 2000), 33-56.

[7] ———, *Shape-Tree Semilattice*, Journal of Mathematical Imaging and Vision **22** (May 2005), 309 - 331.

[8] ———, *Adjacency lattices and shape-tree semilattices, Image & Vision Computing*, Special Issue on ISMM05 **25** (April 2007), no. 4.

[9] P. Monasse, *Morphological representation of digital images and application to registration*, 2000. Ph.D. Thesis, Université Paris IX-Dauphine.

[10] P. Monasse and F. Guichard, *Fast computation of a contrast-invariant image representation*, IEEE Transactions on Image Processing **9** (2000), 860-872.

[11] ———, *Scale-Space from a Level Lines Tree*, Journal of Visual Communication and Image Representation **11** (2000), 224-236.

[12] P. Salembier, *Binary Partition Tree as an Efficient Representation for Image Processing, Segmentation, and Information Retrieval*, IEEE Transactions on Image Processing **9** (April 2000), 561-576.

[13] P. Salembier and L. Garrido, *Connected operators based on region-tree prunning*, 15th International Conference on Pattern Recognition (ICPR'00) **3** (September 2000), 367 - 370.

[14] P. Salembier, L. Garrido, and D. Garcia, *Auto-Dual connected operators based on iterative merging algorithms*, IEEE Transactions on Image Processing **7** (April 1998), no. 4.

[15] P. Salembier, J. Llach, and L. Garrido, *Visual Segment Tree Creation for MPEG-7 Description Schemes*, Pattern Recognition **35** (March 2002), 563-579.

[16] A. Vichik, *Self-dual Morphological Operators Based on Tree Representations of Images*, 2006. Master's Thesis, EE Dept., Technion-IIT, Israel. Available from: <`http://sipl.technion.ac.il/new/Research/Publications/Graduates/Alla_Vichik/Alla_main_revised_f.pdf`>.

[17] S. Vichik, R. Sandler, and A. Rosen, *Moving Car License Plate Recognition*, 1999. Semestrial project, Technion-IIT, Available from: <`www.cs.technion.ac.il/Labs/Isl/Project/Projects_done/cars_plates/finalreport.htm`>.

II

GEOMETRY AND TOPOLOGY

Locally finite spaces and the join operator

Erik Melin

Department of Mathematics, Uppsala University, Sweden
melin@math.uu.se

Abstract The importance of digital geometry in image processing is well documented. To understand global properties of digital spaces and manifolds we need a solid understanding of local properties. We shall study the join operator, which combines two topological spaces into a new space. Under the natural assumption of local finiteness, we show that spaces can be uniquely decomposed as a join of indecomposable spaces.

Keywords: join operator, Alexandrov space, smallest-neighborhood space, locally finite space.

1. Introduction

Topological properties of digital images play an important role in image processing. Much of the theoretical development has been motivated by the needs in applications, for example digital Jordan curve theorems and the theory of digitization. A classical survey is [11] by Kong and Rosenfeld.

Inspired by the new mathematical objects that have emerged from this process, mathematicians have started to study digital geometry from a more theoretical perspective, developing the theories in different directions. Evako et al. [5, 6] considered, for example, n-dimensional digital surfaces satisfying certain axioms. These surfaces were later considered by Daragon et al. [4]. Khalimsky spaces as surfaces, embedded in spaces of higher dimension have been studied the author in [15].

We shall study, not digital spaces, but a tool that can be used in such a study: the join operator. Evako used an operation on directed graphs called the join to aid the analysis. We will generalize this construction and study its properties. In particular we show how a space can be decomposed into indecomposable pieces put together by the join operator. We will give conditions for uniqueness of this decomposition.

2. Digital spaces and the Khalimsky topology

We present here a mathematical background. The purpose is primarily to introduce notation and formulate some results that we will need. A reader not familiar with these concepts is recommended to take a look at, for example, Kiselman's [9] lecture notes.

2.1 Topology and smallest-neighborhood spaces

Not all topological spaces are reasonable digital spaces, but the class of finite topological spaces is too small. It does not include $\mathbb{Z}^n$.

In every topological space, a *finite* intersection of open sets is open, whereas an *arbitrary* intersection of open sets need not be open. If the space is finite, however, there are only finitely many open sets, so finite spaces fulfill a stronger requirement: arbitrary intersections of open sets are open.

Alexandrov [1] considers topological spaces, finite or not, that fulfill this stronger requirement. We shall call such spaces *smallest-neighborhood spaces*. Another name that is often used is *Alexandrov spaces*, but this name has one disadvantage: it has already been used for spaces appearing in differential geometry.

Let B be a subset of a topological space X. The *closure* of B is the intersection of all closed sets containing B. The closure is usually denoted by $\overline{B}$. We shall instead write $\mathscr{C}_X(B)$ for the closure of B in X. This notation allows us to specify in what space we consider the closure and is also a notation dual to $\mathscr{N}_X$ defined below.

Using the same B and X as above, we define $\mathscr{N}_X(B)$ to be the intersection of all open sets containing B. In general $\mathscr{N}_X(B)$ is not an open set, but in a smallest-neighborhood space it is; $\mathscr{N}_X(B)$ is the smallest neighborhood containing B. If there is no danger of ambiguity, we will just write $\mathscr{N}(B)$ and $\mathscr{C}(B)$ instead of $\mathscr{N}_X(B)$ and $\mathscr{C}_X(B)$. If x is a point in X, we define $\mathscr{N}(x) = \mathscr{N}(\{x\})$ and $\mathscr{C}(x) = \mathscr{C}(\{x\})$. Note that $y \in \mathscr{N}(x)$ if and only if $x \in \mathscr{C}(y)$.

We have already seen that $\mathscr{N}(x)$ is the smallest neighborhood of x. Conversely, if every point of the space has a smallest neighborhood, then an arbitrary intersection of open sets is open; hence this existence could have been used as an alternative definition of a smallest-neighborhood space.

A point x is called *open* if $\mathscr{N}(x) = \{x\}$, that is, if $\{x\}$ is open. The point is called *closed* if $\mathscr{C}(x) = \{x\}$. If x is either open or closed it is called *pure*, otherwise it is called *mixed*.

Adjacency and connectedness

A topological space X is called *connected* if the only sets, which are both closed and open, are the empty set and X itself. A *connectivity component* (sometimes called a "connected component") of a topological space is a connected subspace which is maximal with respect to inclusion.

Two distinct points x and y in X are called *adjacent* if the subspace $\{x, y\}$ is connected. It is easy to check that x and y are adjacent if and only $y \in \mathscr{N}(x)$ or $x \in \mathscr{N}(y)$. Another equivalent condition is $y \in \mathscr{N}(x) \cup \mathscr{C}(x)$. The *adjacency neighborhood* of a point x in X is denoted $\mathrm{AN}_X(x)$ and is the set $\mathscr{N}_X(x) \cup \mathscr{C}_X(x)$. It is practical also to have a notation for the set of points adjacent to a point, but not including it. Therefore the *adjacency set* in X of a point x, denoted $\mathscr{A}_X(x)$, is defined to be $\mathscr{A}_X(x) = \mathrm{AN}_X(x) \smallsetminus \{x\}$. Often, we just write $\mathscr{A}(x)$ and $\mathrm{AN}(x)$.

A point adjacent to x is sometimes called a *neighbor* of x. This terminology, however, is somewhat dangerous since a neighbor of x need not be in the smallest neighborhood of x.

Separation axioms

Kolmogorov's separation axiom, also called the T_0 axiom, states that given two distinct points x and y, there is an open set containing one of them but not the other. An equivalent formulation is that $\mathscr{N}(x) = \mathscr{N}(y)$ implies $x = y$ for every x and y. The $T_{1/2}$ axiom states that all points are pure. Clearly any $T_{1/2}$ space is also T_0.

The next separation axiom is the T_1 axiom. It states that points are closed. In a smallest-neighborhood space this implies that every set is closed and hence that every set is open. Therefore, a smallest-neighborhood space satisfying the T_1 axiom must have the discrete topology, and thus, is not very interesting.

Duality

Since the open and closed sets in a smallest neighborhood space X satisfy exactly the same axioms, there is a complete symmetry. Instead of calling the open sets open, we may call them closed, and call the closed sets open. Then we get a new smallest-neighborhood space, called the *dual* of X, which we will denote by X'.

The Alexandrov–Birkhoff preorder

There is a correspondence between smallest-neighborhood spaces and partially preordered sets. Let X be a smallest-neighborhood space and define $x \preccurlyeq y$ to hold if $y \in \mathscr{N}(x)$. We shall call this relation the *Alexandrov–Birkhoff preorder.* It was studied independently by Alexandrov [1] and by Birkhoff [3].

The Alexandrov–Birkhoff preorder is always reflexive (for all x is $x \preccurlyeq x$) and transitive (for all $x, y, z \in X$, $x \preccurlyeq y$ and $y \preccurlyeq z$ imply $x \preccurlyeq z$). A relation satisfying these conditions is called a *preorder* (or *quasiorder*).

A preorder is an *order* if it in addition is anti-symmetric (for all $x, y \in X$, $x \preccurlyeq y$ and $y \preccurlyeq x$ imply $x = y$). The Alexandrov–Birkhoff preorder is an order if and only the space is T_0. In conclusion, it would therefore be possible to formulate the results of this paper in the language of orders instead of the language of topology.

2.2 The Khalimsky topology

We shall construct a connected topology on $\mathbb{Z}$, which was introduced by Efim Khalimsky (see Khalimsky et al. [8] and references there).

If m is an odd integer, let $P_m =]m-1, m+1[$, and if m is an even integer, let $P_m = \{m\}$. The family $\{P_m\}_{m\in\mathbb{Z}}$ forms a partition of the Euclidean line $\mathbb{R}$ and thus we may consider the quotient space. If we identify each P_m with the integer it contains, we get the *Khalimsky topology* on $\mathbb{Z}$. We call this space the *Khalimsky line.* Since $\mathbb{R}$ is connected, the Khalimsky line is a connected space.

It follows readily that an even point is closed and that an odd point is open. In terms of smallest neighborhoods, we have $\mathscr{N}(m) = \{m\}$ if m is odd and $\mathscr{N}(n) = \{n-1, n, n+1\}$ if n is even.

Perhaps this topology should instead be called the Alexandrov–Hopf–Khalimsky topology, since it appeared in an exercise [2, I:Paragraph 1:Exercice 4]. However, it was Khalimsky who realized that this topology was useful in connection with digital geometry and studied it systematically. Since this topology is also called the Khalimsky topology in the literature, we will keep this name.

A different approach to digital spaces, using cellular complexes, was introduced independently by Herman and Webster [7] and by Kovalevsky [13]. The results of this article apply to such spaces, since they are topologically equivalent to spaces of the type introduced in this article. See, for example, Klette [10].

Khalimsky intervals and arcs

Let a and b, $a \leqslant b$, be integers. A *Khalimsky interval* is an interval $[a, b] \cap \mathbb{Z}$ of integers with the topology induced from the Khalimsky line. We will denote such an interval by $[a, b]_{\mathbb{Z}}$ and call a and b its *endpoints.* A *Khalimsky arc* in a topological space X is a subspace that is homeomorphic to a Khalimsky interval. If any two points in X are the endpoints of a Khalimsky arc, we say that X is *Khalimsky arc-connected.*

Theorem 1. *A T_0 smallest-neighborhood space is connected if and only if it is Khalimsky arc-connected.*

A proof can be found in [14, Theorem 11]. Slightly weaker is [8, Theorem 3.2c]. The theorem also follows from Lemma 20(b) in [12].

Let us define the *length* of a Khalimsky arc, A, to be the number of points in A minus one, $L(A) = \operatorname{card} A - 1$. If a smallest-neighborhood space X is T_0 and connected, Theorem 1 guarantees that length of the shortest arc connecting x and y in X is a finite number. This observation allows us to define a metric ρ_X on X, which we call the *arc metric.*

$$\rho_X(x, y) = \min(L(A);\ A \subset X \text{ is a Khalimsky arc containing } x \text{ and } y).$$

The arc metric is defined on X, but it is important to bear in mind that the topology of X is not the metric topology defined by ρ_X. The metric topology is of course the discrete topology.

Examples of smallest-neighborhood spaces

We conclude this section with a few examples to indicate that the class of smallest-neighborhood spaces and spaces based on the Khalimsky topology is sufficiently rich to be worth studying.

Example 1. The *Khalimsky plane* is the Cartesian product of two Khalimsky lines and in general, *Khalimsky n-space* is $\mathbb{Z}^n$ with the product topology. Points with all coordinates even are closed and points with all coordinates odd are open. Points with both even and odd coordinates are mixed. It is easy to check that $\mathscr{A}(p) = \{x \in \mathbb{Z}^n;\ \|p - x\|_\infty = 1\}$ if p is pure.

Example 2. We may consider a quotient space $\mathbb{Z}_m = \mathbb{Z}/m\mathbb{Z}$ for some even integer $m \geqslant 2$. Such a space is called a *Khalimsky circle.* If $m \geqslant 4$, $\mathbb{Z}_m$ is a compact space that is locally homeomorphic to the Khalimsky line. (If m were odd, we would identify open and closed points, resulting in a space with the indiscrete or chaotic topology, i.e., where the only open sets are the empty set and the space itself.)

3. Locally finite and locally countable spaces

A topological space actually stored in a computer is finite. Nevertheless, it is of theoretical importance to be able to treat infinite spaces, like $\mathbb{Z}^2$, since the existence of a boundary often tends to complicate matters.

On the other hand, spaces where a points can have an infinite number of neighbors seems less likely to appear in computer applications. In this situation, not even the local information can be stored.

Definition 1. A smallest-neighborhood space is called *locally finite* if every point in it has a finite adjacency neighborhood. If every point has a countable adjacency neighborhood, it is called *locally countable.*

We need to assume the axiom of choice (in fact the countable axiom of choice, which states that from a *countable* collection of nonempty sets we can select one element from each set, is sufficient) to prove the following.

Proposition 1. *Let X be a locally finite space. If X is connected, then X is countable.*

Proof. If X is not T_0, we consider the quotient space $\tilde{X}$, where points with identical neighborhoods have been identified. $\tilde{X}$ is T_0 and X is countable if $\tilde{X}$ is countable. It is therefore sufficient to prove the result for T_0 spaces.

Let x be any point in X. By induction based on local finiteness, the ball $B_n(x) = \{y; \in X; \rho_X(x,y) \leqslant n\}$, where ρ_X is the arc-metric on X (here we use that the space is T_0), is finite for every $n \in \mathbb{N}$. Since $X = \bigcup_{n=0}^{\infty} B_n(x)$, the result is true, since the countable axiom of choice implies that a countable union of finite sets is countable. □

In a similar way, we can also characterize countable smallest-neighborhood spaces.

Proposition 2. *A smallest-neighborhood space X is countable if and only if it is locally countable and has countably many connectivity components.*

Proof. It is clear that a countable space is locally countable and has countably many components. The countable axiom of choice implies that countable unions of countable sets are countable. So if X is connected and locally countable, a slightly modified version of the proof of Proposition 1 shows that X is countable. If X has countably many connectivity components and each component is countable, then X is countable. □

The following proposition states that the set of open points is dense in a locally finite space (and by duality a corresponding result holds for the closed points). It is obvious that the open points form the smallest dense set.

Proposition 3. *Let X be a smallest-neighborhood space and let $S \subset X$ be the set of open points in X and T be the set of closed points. If X is T_0 and locally finite, then $X = \mathscr{C}(S) = \mathscr{N}(T)$.*

Proof. We will prove the fist equality, the other follows by duality. Let y_0 be any point in X. Let $Y_0 = \mathscr{N}(y_0)$. If Y_0 is a singleton set, then $y_0 \in S$. Otherwise, for $k \geqslant 0$, choose $y_{k+1} \in Y_k \smallsetminus \{y_k\}$ and let $Y_{k+1} = \mathscr{N}(y_{k+1})$. Clearly $Y_{k+1} \subset Y_k$ and since X is T_0, it follows that $y_k \notin Y_{k+1}$. Repeat the construction above recursively until Y_k is a singleton set, at most $\operatorname{card}(Y_0) - 1$ steps are needed. Note that y_k is an open point and that $y_0 \in \mathscr{C}(y_k)$. Since y_0 was arbitrarily chosen, we are done. □

The relation in the proposition need not hold if the space is not locally finite. Consider the space $\mathbb{Z}$ where the (non-trivial) open sets are given by intervals $]-\infty, m]_{\mathbb{Z}}$, $m \in \mathbb{Z}$. This space contains no open point.

On the other hand, if we add the point $-\infty$ and declare the open sets to be intervals $[-\infty, m]_{\mathbb{Z}}$, where $m \in \mathbb{Z} \cup \{-\infty\}$, then the point $-\infty$ is open and the whole space equals $\mathscr{C}(-\infty)$. Thus, finite neighborhoods are not necessary for the conclusion of the proposition.

4. The join operator

A well-known way to combine two topological spaces, X and Y is to take the coproduct (disjoint union), $X \coprod Y$. The pieces, X and Y, are completely independent in this construction. We shall introduce another way of combining two spaces, namely the join operator.

Definition 2. Let X and Y be two topological spaces. The *join* of X and Y, denoted $X \vee Y$, is a topological space over the disjoint set union of X and Y, where a subset $A \subset X \dot{\cup} Y$ is declared to be open if either

(i) $A \cap X$ is open in X and $A \cap Y = \emptyset$, or

(ii) $A \cap X = X$ and $A \cap Y$ is open in Y.

Note that a set $B \subset X \vee Y$ is closed if and only if

(i) $B \cap X$ is closed in X and $B \cap Y = Y$, or

(ii) $B \cap X = \emptyset$ and $B \cap Y$ is closed in Y.

While this definition makes sense for any topological space, it is a strange operation on large spaces. For example, the join of the real line and the circle, $\mathbb{R} \vee S^1$, is a compact space, which is T_0 but not T_1. In fact, $X \vee Y$ cannot be T_1 unless X or Y is empty.

From now on, we shall only consider the join of smallest-neighborhood spaces. In this case, the definition boils down to the following. If X and Y are smallest-neighborhood spaces, then the topology of $X \vee Y$ is given by $\mathscr{N}_{X \vee Y}(x) = \mathscr{N}_X(x)$ if $x \in X$ and $\mathscr{N}_{X \vee Y}(y) = \mathscr{N}_Y(y) \cup X$ if $y \in Y$. Apparently, the join of two smallest-neighborhood spaces is a smallest-neighborhood space. This definition of the join is compatible with the join of directed graphs, see [6, p. 111]. In terms of the Alexandrov–Birkhoff preorder, every element of X is declared to be larger than any element of Y; X is placed on top of Y in $X \vee Y$. This motivates also our notation $X \vee Y$. Formally, the order, i.e., pairs (x, y) satisfying $x \succcurlyeq y$, on $X \vee Y$, which we here denote $\operatorname{Ord}(X \vee Y)$ is

$$\operatorname{Ord}(X \vee Y) = \operatorname{Ord}(X) \cup \operatorname{Ord}(Y) \cup X \times Y.$$

The join of two connected locally finite spaces need not be locally finite. In fact, the join of two spaces is locally finite if and only if the spaces are finite and locally countable if and only if both are countable. In view of Proposition 2, the join of two connected and locally countable spaces is locally countable. When p is a point, we write $p \vee X$ instead of $\{p\} \vee X$. Here $\{p\}$ is the topological space with one point.

4.1 Basic properties

The following three properties in the next proposition are straightforward to prove.

Proposition 4. *The join operator has the following properties for all smallest-neighborhood spaces X, Y and Z.*

(i) $X = \emptyset \vee X = X \vee \emptyset$ *(has a unity).*

(ii) $(X \vee Y) \vee Z = X \vee (Y \vee Z)$ *(is associative).*

(iii) $(X \vee Y)' = Y' \vee X'$.

The following proposition lists some topological properties.

Proposition 5. *Let X and Y be smallest-neighborhood spaces.*

(i) *$X \vee Y$ is T_0 if and only if X and Y are T_0.*

(ii) *$X \vee Y$ is compact if and only if Y is compact.*

(iii) *If $X \neq \emptyset$ and $Y \neq \emptyset$ then $X \vee Y$ is connected.*

Proof. To prove (i), note that X is open in $X \vee Y$. If $x \in X$ and $y \in Y$, then X is an open set containing x but not y. It follows that $X \vee Y$ can fail to be T_0 only for a pair of points in X or a pair of points in Y. But in this case the equivalence is obvious.

Next we prove (ii). Assume first that Y is not compact and that $\{A_i\}_{i \in I}$ is an open cover of Y without a finite subcover. Let $B_i = A_i \cup X$ for each $i \in I$. Then $\{B_i\}_{i \in I}$ is an open cover of $X \vee Y$ without a finite subcover. For the other direction, assume that Y is compact and take an open cover, $\{B_i\}_{i \in I}$, of $X \vee Y$. By restriction, it induces an open cover of Y with elements $B_i \cap Y$. But this cover has a finite subcover, $\{B_i \cap Y\}_{i=1}^{n}$, since Y is compact. It follows that $\{B_i\}_{i=1}^{n}$ is finite subcover of $X \vee Y$ since any B_i where $B_i \cap Y \neq \emptyset$ covers X.

To prove (iii), assume that $x \in X$ and $y \in Y$. Since $x \in \mathscr{N}(y)$, it is clear that x and y are adjacent. If $a, b \in X$, then $\{a, y, b\}$ a connected set for the same reason. In the same way $\{c, x, d\}$ is connected if $c, d \in Y$. □

In fact all properties of the two proceeding propositions, except (iii) of Proposition 4, are true for general topological spaces, not only for smallest-neighborhood spaces.

4.2 Decomposable spaces

If $Z = X \vee Y$ implies $X = \emptyset$ or $Y = \emptyset$, then the smallest-neighborhood space Z is called *indecomposable*, otherwise Z is called *decomposable.* Note that a locally finite decomposable smallest-neighborhood space is in fact finite.

Proposition 6. *If a smallest-neighborhood space Z is decomposable, then for every $x, y \in Z$ there is a point $z \in Z$ such that $x, y \in \mathrm{AN}(z)$. (If Z is T_0, an equivalent and more comprehensible condition is that $\rho_Z(x, y) \leqslant 2$.)*

Proof. The result is given by the proof of Proposition 5, Part (iii). □

The converse implication is not true, as the following example shows.

Example 3. Let $X = \{a_1, a_2, b, c_1, c_2\}$ be a set with 5 points, and equip it with a topology as follows: $\mathscr{N}(a_1) = \{a_1\}$, $\mathscr{N}(a_2) = \{a_2\}$, $\mathscr{N}(b) = \{a_1, b\}$, $\mathscr{N}(c_1) = \{a_1, a_2, b, c_1\}$, and $\mathscr{N}(c_2) = \{a_1, a_2, c_2\}$. It is easy to see that $\rho(x, y) \leqslant 2$ for any $x, y \in X$.

Suppose that X were decomposable, $X = A \vee C$. We would necessarily have $a_1, a_2 \in A$ since these points are open, and $c_1, c_2 \in C$ since these points are closed. But b cannot be in A since $b \notin \mathscr{N}(c_2)$ and b cannot be in C since $a_2 \notin \mathscr{N}(b)$. Therefore, X is indecomposable.

We have the following uniqueness result for the decomposition.

Theorem 2. *Let X be a smallest-neighborhood space. If $X = Y \vee Z$ and $X = \tilde{Y} \vee \tilde{Z}$, where Y and $\tilde{Y}$ are indecomposable and non-empty, then $Y = \tilde{Y}$ and $Z = \tilde{Z}$.*

Proof. It is sufficient to prove that $Y = \tilde{Y}$. If $Y \neq \tilde{Y}$, we may suppose that $Y \smallsetminus \tilde{Y}$ is not empty and let $p \in Y \smallsetminus \tilde{Y}$. Then $\tilde{Y} \subset Y$, for if there were a point q in $\tilde{Y} \smallsetminus Y$, then $q \in Z$ so that $\mathscr{C}_X(q) \subset Z$, since $X = Y \vee Z$. But by assumption $p \notin \tilde{Y}$, so therefore $p \in \tilde{Z}$. As $q \in \tilde{Y}$ and $X = \tilde{Y} \vee \tilde{Z}$, this implies $p \in \mathscr{C}_X(q) \subset Z$, which is a contradiction since $p \in Y$.

Define $B = Y \smallsetminus \tilde{Y}$, which is non-empty by assumption. Take two arbitrary points $a \in \tilde{Y}$ and $b \in B$. Note that $b \in \tilde{Z}$. Hence $a \in \mathscr{N}_X(b)$ and thus also $a \in \mathscr{N}_Y(b)$. It follows that $\mathscr{N}_Y(b) = \mathscr{N}_B(b) \cup \tilde{Y}$. If $\tilde{Y}$ and B are equipped with the relative topology, we have $Y = \tilde{Y} \vee B$, so Y is decomposable contrary to the assumption. $\square$

If a smallest-neighborhood space X is locally finite, then repeated use of Theorem 2 together with associativity gives the following.

Corollary 1. *If X is a locally finite smallest-neighborhood space, then X can be written in a unique way as $X = Y_1 \vee \cdots \vee Y_n$ where each Y_i is indecomposable and non-empty.*

Note that if X is decomposable so that $n > 1$, then X is necessarily finite. While Theorem 2 is quite general, there is no universal cancellation law; if $A \vee X = B \vee X$ we cannot conclude that A and B are homeomorphic (which we will denote by $A \simeq B$), as the following example shows.

Example 4. Let N denote the set of natural numbers equipped with the topology given by $\mathscr{N}(n) = \{i \in \mathbb{N};\ i \geqslant n\}$. For a positive integer m, let $N_m = N \cap [0, m]$, with the induced topology. It follows that $N = N_m \vee N$ for every m, but N_m is homeomorphic to N_k only if $m = k$.

On the other hand, if X is locally finite, the unique decomposition of Corollary 1 implies that both a right and a left cancellation law hold. In fact, we may weaken the hypothesis slightly.

Theorem 3. *Let X be a smallest-neighborhood space. Suppose there are finitely many locally finite spaces $Y_1, \ldots, Y_n$ so that $X = Y_1 \vee \cdots \vee Y_n$. Then for all smallest-neighborhood spaces A and B we have*

(i) $X \vee A \simeq X \vee B$ *implies* $A \simeq B$,

(ii) $A \vee X \simeq B \vee X$ *implies* $A \simeq B$.

Note that X is locally finite only if $n = 1$ or if every Y_i (and hence X itself) is finite.

Proof. We shall prove (i). The second claim follows by duality. Let Y be any smallest-neighborhood space. Define an *opening chain* in Y to be a finite sequence of pairwise distinct points $(y_0, \dots, y_n)$ in Y such that $y_{i+1} \in \mathscr{N}_Y(y_i)$ for $0 \leqslant i \leqslant n$. The number n is the called the *length* of the open chain. We say that the chain *starts* in y_0. Let $h_Y : Y \to \mathbb{N} \cup \{\infty\}$ be defined by

$$h_Y(y) = \sup(n; \text{ there is an opening chain in } Y \text{ of length } n \text{ starting in } y).$$

From the construction of X, it is straightforward to check that $h_{A\vee X}(x)$ is a finite number for any $x \in X$. Furthermore, for any $a \in A$ we have

$$h_{X\vee A}(a) \geqslant 1 + \sup_{x\in X} h_{X\vee A}(x) \tag{1}$$

since every $x \in X$ belongs to $\mathscr{N}_{X\vee A}(a)$. Furthermore,

$$\sup_{x\in X} h_{X\vee A}(x) = \sup_{x\in X} h_{X\vee B}(x), \tag{2}$$

since $h_X(x) = h_{X\vee Y}(x)$ for any space Y and any $x \in X$.

Let $\varphi \colon X \vee A \to X \vee B$ be a homeomorphism. A chain is mapped to a chain by φ, so we have the identity $h_{X\vee A}(x) = h_{X\vee B}(\varphi(x))$ for every $x \in X \vee A$. In view of (1) and (2) this implies that

$$h_{X\vee B}(\varphi(a)) \geqslant 1 + \sup_{x\in X} h_{X\vee B}(x),$$

for every $a \in A$. Hence $\varphi(X) = X$ and $\varphi(A) = B$, which proves (i). □

5. Applications

We shall demonstrate how the tools we have developed can be used to give a simple proof of a known result in digital topology, namely the characterization of neighborhoods in Khalimsky spaces (Evako et al. [6]). We start with a consequence of Proposition 6.

Corollary 2. *Let $p \in \mathbb{Z}^n$ be pure. Then $\mathscr{A}(p)$ is indecomposable. If p is closed,* $\mathrm{AN}(p) = \mathscr{A}(p) \vee p$ *and if p is open,* $\mathrm{AN}(p) = p \vee \mathscr{A}(p)$.

Proof. For the first property, notice that if $q \in \mathscr{A}(p)$ then $r = 2p - q$ (as vectors in $\mathbb{Z}^n \subset \mathbb{R}^n$) also belongs to $\mathscr{A}(p)$. It is readily checked that $\rho_{\mathscr{A}(p)}(q, r) = 4$ if $n \geqslant 2$ (if $n = 1$, $\mathscr{A}_{\mathbb{Z}}(p)$ is not connected and the result immediate). Proposition 6 shows that $\mathscr{A}(p)$ is indecomposable. The decomposition of $\mathrm{AN}(p)$ is straightforward. □

If p is pure, it is easy to explicitly describe $\mathrm{AN}(p)$. We have

$$\mathrm{AN}(p) = \{x \in \mathbb{Z}^n;\ \|x - p\|_\infty \leqslant 1\},$$

so that p is adjacent to $3^n - 1$ points.

More generally, let $q \in \mathbb{Z}$ be any point with j even coordinates and k odd coordinates. To simplify our notation, we note that there is a homeomorphism of $\mathbb{Z}^n$, build from a translation $q \mapsto q + v$, where $v \in 2\mathbb{Z}^n$, and permutation of coordinates, which takes q to the point $\tilde{q} = (0, \ldots, 0, 1, \ldots 1) \in \mathbb{Z}^n$, where there are j zeros and k ones (and $j + k = n$).

It follows that

$$\mathscr{N}(\tilde{q}) = [-1, 1]^j \times \{1\}^k$$

and that

$$\mathscr{C}(\tilde{q}) = \{0\}^j \times [0, 2]^k.$$

Since $\mathrm{AN}(\tilde{q}) = \mathscr{N}(\tilde{q}) \cup \mathscr{C}(\tilde{q})$, we see that q is adjacent to $3^j + 3^k - 2$ points.

Let $\mathbf{0}_j$ denote the point $(0, \ldots, 0) \in \mathbb{Z}^j$ and let $\mathbf{1}_k = (1, \ldots, 1) \in \mathbb{Z}^k$. It is easy to see that $\mathscr{N}(\tilde{q}) \simeq \mathscr{N}_{\mathbb{Z}^j}(\mathbf{0}_j)$ and it follows that $\mathscr{N}(\tilde{q}) \smallsetminus \{\tilde{q}\}$ is homeomorphic to $\mathscr{A}_{\mathbb{Z}^j}(\mathbf{0}_j)$, which is indecomposable by Corollary 2. By a similar argument, $\mathscr{C}(q) \smallsetminus \{\tilde{q}\}$ is homeomorphic to $\mathscr{A}_{\mathbb{Z}^k}(\mathbf{1}_k)$. Note that $\mathscr{A}(\mathbf{0}_0) = \mathscr{A}(\mathbf{1}_0) = \emptyset$.

It is straightforward to check that in any smallest-neighborhood space X and for any $x \in X$ we have

$$\mathscr{A}(x) \simeq (\mathscr{N}(x) \smallsetminus \{x\}) \vee (\mathscr{C}(x) \smallsetminus \{x\})$$

and we obtain the following.

Proposition 7. *Let $q \in \mathbb{Z}^n$. Then*

$$\mathscr{A}_{\mathbb{Z}^n}(q) \simeq \mathscr{A}_{\mathbb{Z}^j}(\mathbf{0}_j) \vee \mathscr{A}_{\mathbb{Z}^k}(\mathbf{1}_k),$$

where j is the number of even coordinates in q and k is the number of odd coordinates.

As stated from the outset, this result is known, and serves only as an illustration of the formalism introduced.

6. Conclusion

We have studied the join operator, which takes two topological spaces and combines them into a new space. This operation is interesting primarily for small spaces, viz. adjacency neighborhoods. We have seen that if we assume local finiteness of the spaces involved, we can show that a space is decomposed in a unique way into indecomposable spaces, and we have given a criterion to recognize indecomposable spaces.

The machinery can be used to systematically investigate local properties of digital topological spaces. Hopefully, this will lead to new insights into the nature of such spaces.

References

[1] P. Alexandrov, *Diskrete Räume*, Mat. Sb. **2** (1937), no. 44, 501–519.

[2] P. Alexandrov and H. Hopf, *Topologie I*, Julius Springer, Berlin, 1935.

[3] G. Birkhoff, *Rings of sets*, Duke Math J. **3** (1937), no. 3, 443–454.

[4] X. Daragon, M. Couprie, and G. Bertrand, *Derived neighborhoods and frontier orders*, Discrete Appl. Math. **147** (2005), no. 2-3, 227–243.

[5] A. V. Evako, *Dimension on discrete spaces*, Internat. J. Theoret. Phys. **33** (1994), no. 7, 1553–1567.

[6] A. V. Evako, R. Kopperman, and Y. V. Mukhin, *Dimensional properties of graphs and digital spaces*, J. Math. Imaging Vision **6** (1996), no. 2–3, 109–119.

[7] G. T. Herman and D. Webster, *A topological proof of a surface tracking algorithm*, Computer Vision, Graphics, and Image Processing **23** (1983), 162–177.

[8] E. Khalimsky, R. Kopperman, and P. R. Meyer, *Computer graphics and connected topologies on finite ordered sets*, Topology Appl. **36** (1990), no. 1, 1–17.

[9] C. O. Kiselman, *Digital geometry and mathematical morphology*, Uppsala University, 2004. Available at `www.math.uu.se/~kiselman`.

[10] R. Klette, *Topologies on the planar orthogonal grid*, 6th International Conference on Pattern Recognition (ICPR'02), 2002, pp. 354–357.

[11] T. Y. Kong and A. Rosenfeld, *Digital topology: Introduction and survey*, Comput. Vision Graph. Image Process. **48** (1989), no. 3, 357–393.

[12] R. Kopperman, *The Khalimsky line as a foundation for digital topology*, Shape in picture, 1994, pp. 3–20.

[13] V. A. Kovalevsky, *Finite topology as applied to image analysis.*, Comput. Vision Graph. Image Process. **46** (1989), 141–161.

[14] E. Melin, *Continuous extension in topological digital spaces*, Technical Report 2004:2, Uppsala University, 2004. To appear in *Applied General Topology.*

[15] ______, *Continuous digitization in Khalimsky spaces*, Technical Report 2006:3, Uppsala University, 2006. To appear in *J. Approx. Theory.*

Digital Steiner sets and Matheron semi-groups

JEAN SERRA

Laboratoire L2ASI, ESIEE, France
serraj@esiee.fr

Abstract The Euclidean hirerachies of openings satisfy Matheron semi-groups law $\gamma_\lambda\gamma_\mu = \gamma_{\max(\lambda,\mu)}$, where λ is a size factor. One finds this law when the γ_λ are adjunction openings by Steiner convex sets, i.e. by Minkowski sums of segments. The conditions under which, in Z^n, the law remains valid, and the Steiner sets are convex, and connected, are established.

Keywords: Matheron semi-group, granulometry, digital, convexity, Steiner, connection, connectivity.

1. Matheron semi-groups and convexity

In the practice of morphological image processing, one often uses families of mappings that depend on a positive factor λ which expresses a size. When the mappings are idempotent, a convenient model for these hierarchies is the *semi-group* introduced by G. Matheron [6] Ch. 7, as a *Euclidean granulometry*, where the γ_λ's are openings such that

$$\gamma_\lambda\gamma_\mu = \gamma_{\max(\lambda,\mu)} \qquad \lambda \geq \mu \geq 0. \tag{1}$$

As a matter of fact, this law is associated with many other idempotent operators, such as the alternating sequential filters [13], or with the levelings [14]. Now all these filters derive from the two basic types of the opening by adjunction and the connected opening.

We propose to study here the discrete version of Matheron semi-groups (1) for the openings by adjunction. It is known that in the Euclidean case, these operators lie on *convex* structuring elements [6], which are the more often obtained by Minkowski sum of segments in different directions. They are then called Steiner compact sets (see Definition 1), and coincide in R^2 with the compact convex sets with a center of symmetry. Now in Z^n the Minkowski sum of two segments may be not convex, and symmetrical convex sets may not be Steiner (Figure 4(a)). Must we renounce discrete granulometries by adjunction, or deduce that convexity plays no role in digital Matheron semi-groups? Here is the first question we have to clear up.

But it is not the only one. In vector spaces, compact convex sets are equivalently defined by barycentres, or by intersection of half-spaces. This is no longer true in Z^n (Figure 4(a)). But what can mean "a straight line segment", or "a half-space", in Z^n? Must we choose among several definitions? Are they definitions for which both convexities are the same? If so, do they lead to nice hierarchies? On the other hand, what about connectivity? In R^n, but not in Z^n, convex sets are always connected (Figure 4(a)). Is it a handicap?

In digital geometry, one may either consider the module Z^n as a part of the vector space R^n, or the latter as a possible generalization of module Z^n. In our case, since we deal with structuring elements, i.e., with (necessarily discrete) actions on the objects and not with the objects themselves, the Z^n framework turns out to be the convenient one.

2. Reminders

Symbol $\mathcal{L}$ indicates a complete lattice, whose elements are denoted by capital letters. When $\mathcal{L}$ is of $\mathcal{P}(E)$ type, the elements of E are given by lower case letters. Let $\mathcal{L}$ admit a class $\mathcal{S}$ of sup-generators, and let $\{\delta_\lambda\}, \lambda \geq 0$ be a family of dilations on $\mathcal{L}$, of adjoint erosions $\{\varepsilon_\lambda\}$. It is known [13, 20] that the family of the openings by adjunction $\{\gamma_\lambda = \delta_\lambda \varepsilon_\lambda\}, \lambda \geq 0$, then forms a granulometry if and only if we have for all $b \in \mathcal{S}$ that

$$\lambda \geq \mu \quad \Rightarrow \quad \delta_\lambda(b) = \gamma_\mu \delta_\lambda(b). \tag{2}$$

In case of $\mathcal{P}(R^n)$, the families of homothetic convex sets are essential [6], because

1. the homothetic version λB of the compact set B is open by μB for all $\lambda \geq \mu$ if and only if B is convex;
2. a family $\{B_\lambda,\ \lambda \geq 0\}$, forms a continuous additive semi-group if and only if B_λ is homothetic of ratio λ of the compact convex set B

$$B_\lambda \oplus B_\mu = B_{\lambda+\mu}\ ,\ \ \lambda, \mu \geq 0 \quad \Leftrightarrow \quad B_\lambda = \lambda B, \quad B_\mu = \mu B, \quad B \text{ convex.} \tag{3}$$

The link between convexity and granulometry by adjunction becomes clear, as applying Equation 2 to Euclidean granulometries, and demanding, in addition, homothetic structuring elements, i.e., $B_\lambda = \lambda B$, yields necessarily to compact convex B's. In other words, for a Euclidean granulometry, the convexity assumption and that of homothetics are equivalent. However, if we relax the magnification assumption, then the $\delta_\lambda(b)$ do not need to be convex, nor even connected.

Among all Euclidean convex sets, the most attractive class turns out to be the *Steiner* one ([6], sect.4.5).

Definition 1. A compact set $K \in \mathcal{K}(R^n)$ is said to be *Steiner* if there exists in $\mathcal{K}$ a sequence $\{K_n\} \in \mathcal{K}$ with $K = Lim K_n$ and if for all $n > 0$, K_n is Minkowski sum of segments centered at a same point. The Steiner class is denoted by $\mathcal{ST}(R^n)$. □

In R^2, Steiner sets are nothing but the symmetrical convex compact ones (rectangle, octagon, etc.) and their limits (discs, ellipses, etc.). In R^n, all 2D faces must be symmetrical convex [15]. The datum of a Steiner set K is equivalent to that of a measure $s_K = s_K(d\alpha)$ on the unit sphere Ω, since

$$K = \oplus\{s_K(d\alpha), \alpha \in \Omega\}. \tag{4}$$

For example, if K is a rectangle, then $s_K(d\alpha)$ reduces to two Dirac measures of orthogonal directions. Equation 4 has for an obvious corollary that the directional measure *exchanges* arithmetic addition and Minkowski one, i.e., $s_{K \oplus K'} = s_K + s_{K'}$, hence

$$s_{K'} \leq s_K \;\Rightarrow\; s_{K \ominus K'} = s_K - s_{K'} \;\Rightarrow\; K \text{ is open by } K'. \tag{5}$$

Consequently, every family of Steiner sets whose directional measures increase generates a granulometry by adjunction.

3. The Z^n module

Which ones of the previous results do remain when R^n is replaced by Z^n? The poorer structure of Z^n is that of a *module*, where (integer) translation is still defined, hence giving access to Minkowski operations. The homothetic factors can only magnify, as they must be integer. Unlike integer translation, which does not pose particular problems, linear equations become a true stumbling block. However, a classical result, due to Bezout, makes precise the existence conditions of solutions.

Proposition 1. *Let $(x_1, x_2..x_n)$ be the coordinates of point $x \in Z^n$. The so called Bezout equation $\sum_1^n a_i x_i = 1$ admits solutions in Z^n if and only if the a_i coefficients relatively prime.*

The point of coordinates $(a_1, a_2...a_n)$ is usually called *Bezout vector.* When one solution $\overrightarrow{u}_0$ Bezout equation $\overrightarrow{a}\overrightarrow{u} = 1$ is known, then the solutions for an arbitrary second member c are given by

$$\overrightarrow{x} = c\overrightarrow{u}_0 + k_1\overrightarrow{w}_1 + ...k_{n-1}\overrightarrow{w}_{n-1} \tag{6}$$

where the $\overrightarrow{w}_1, ...\overrightarrow{w}_{n-1}$ generate the sub-module $A := \{\ \overrightarrow{a}\overrightarrow{x} = 0\}$ of dimension $n-1$. Geometrically, Equation 6 defines a *straight line* in Z^2, a *plane* in Z^3, etc. This geometric structure has the advantage of well scanning Z^n. One goes form the solutions of equation $\overrightarrow{a}\overrightarrow{x} = c$ to those of $\overrightarrow{a}\overrightarrow{x} = c + 1$ by replacing $\overrightarrow{x}$ by $\overrightarrow{x} + \overrightarrow{u}$, where $\overrightarrow{u}$ is an arbitrary solution of Bezout

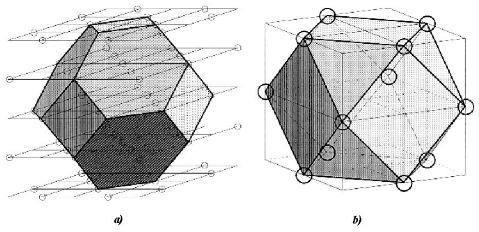


Figure 1. This is a Steiner polyhedron (a), but not that (b).

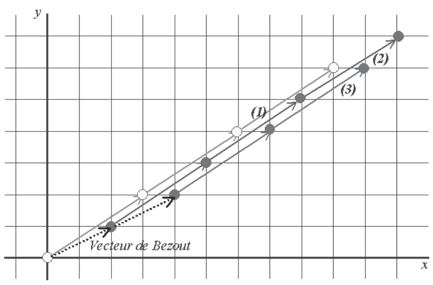


Figure 2. An example of digital plane spanning by Bezout straight lines. Vector (2,1) is a solution of Bezout equation $2x - 3y = 1$, therefore the shifts of this straight line by all multiples of vector (2,1) span integrally the plane.

equation (see Figure 2) so that, as c spans Z, every point of the space Z^n is met once and only once. This nice property is not an exclusivity of the Bezout Straight lines: H. Talbot proved in his Phd thesis that the spanning property is also satisfied by the Bresenham lines [19].

The hyperplanes of the family $\{\sum_1^n a_i x_i = c,\ c \in Z\}$ generate *Bezout half-spaces* $E(A,c) = \sum_1^n a_i x_i \leq c = \cup\{H(A,r), r \leq c\}$, which are nested in each other as c increases.

4. Bezout straight lines and discrete Steiner sets

Since Rosenfeld's pioneer paper [11], digital lines are the matter of an abundant literature, as well as digital convexity. The reader is referred to the survey by Eckardt [2], where at least five different ways for defining digital convexity are distinguished. Most approaches aim to provide digital representations of a Euclidean background, e.g., Rosenfeld for segments [11] or Bresenham for straight lines. Other definitions, such as Reveillès straight lines [10] are introduced in a purely digital framework. More recently, Melin proposed a digital definition in the framework of Khalimsky topology [8].

Kiselman [5] starts from Reveillès digital Equation 11, but immediately reorientates it toward the Euclidian world by making real the integers a_i.

However, the simplest, and above all the narrowest digital straight lines are given by the multiples of Bezout vectors[12, 16]. They will be our starting point.

Definition 2 (Bezout straight lines and segments)**.** Each Bezout vector $\omega = (\omega^1, .., \omega^n)$ defines the *direction* ω in Z^n, the opposite direction $-\omega$ having parameters $(-\omega^1, .., -\omega^n)$. We call a *Bezout straight line* $D(\omega)$, of direction ω and going through the origin, the union of all integer multiples of vector ω, namely $D(\omega) = \{k\omega, \quad k \in Z\}$. Similarly,the Bezout line of direction ω going through point x is written

$$D_x(\omega) = D(\omega) \oplus x = \{x + k\omega, \quad k \in Z\}. \tag{7}$$

Every segment $L_x(k, \omega)$ of $D_x(\omega)$, of origin x and extremity $x + k\omega$, $k \geq 0$, consists in the sequence of the points

$$L_x(k, \omega) = x \cup \{x + p\omega, \quad p \in [0, k]\}, \tag{8}$$

its length is the number $k + 1$ of its points. □

In the following, the Bezout line segments are just called "segment". The set Ω of all directions coincides with Bezout vectors, and corresponds to the unit sphere of the Euclidean case. Note also that, from Equation 7, there exists one and only one Bezout line going through a given point and with a given direction. Minkowski operations for Bezout segments are characterized by the following theorem

Theorem 1. *In $\mathcal{P}(Z^n)$, for all segments $L_x(k, \omega)$ and $L_{x'}(k', \omega)$, $\omega \in \Omega$, $x, x' \in Z^n$, $k, k' \in N$, we have that*

1. *the Minkowski sum $L_x(k, \omega)$ and $L_{x'}(k', \omega)$ is the segment*

$$L_x(k, \omega) \oplus L_{x'}(k', \omega) = L_{x+x'}(k + k', \omega), \tag{9}$$

2. *the Minkowski difference $L_x(k, \omega) \ominus L_{x'}(k', \omega)$ is $L_{x-x'}(k - k', \omega)$ if $k > k'$, $\{x - x'\}$ if $k = k'$ and $= \varnothing$ if $k < k'$,*
3. *the opening of $L_x(k, \omega)$ by $L_{x'}(k', \omega)$ is Segment $L_x(k, \omega)$ itself when $k \geq k'$ or the empty set when not.*

Conversely, the first property is only satisfied by the Bezout segments, and their periodic sub-sets, to the exclusion of the segments of any other straight line with a finite thickness.

Proof. The three properties derive from Definition 2 in the same manner. For the first one, for example, it suffices to write the Minkowski addition

$$L_x(k, \omega) \oplus L_{x'}(k', \omega) = \{x + x'\} \cup \{x + x' + (p + p')\omega, \quad p \in [1, k], \ p \in [1, k']\}$$

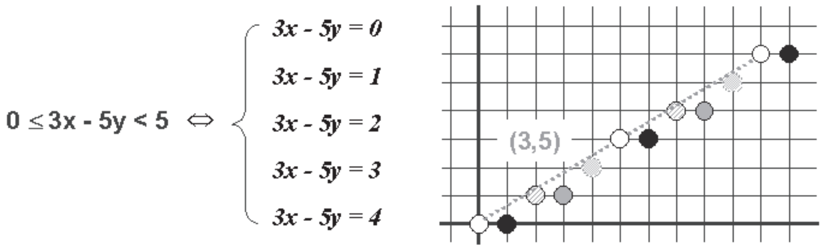


Figure 3. Decomposition of the naive Réveillès straight line $D = \{0 \leq 3x - 5y < 5\}$ into five Bezout lines.

to find Equation 9.

Conversely, suppose Equation 9 satisfied, and consider a set Δ^* that contains point x. If, as y and y' span Δ^* vector $\overrightarrow{yy'}$ always keeps the same direction, the latter can only be a multiple of a Bezout direction ω, so that y and y' describe a periodic sub-set Δ^* of the Bezout line $\Delta_x(\omega)$. If not, put the origin x in a point of Δ^* where both length direction ω and thickness direction ω' coexist. One can always find two segments $L_0(k, \omega)$ and $L_0(k', \omega')$ in Δ^*. By dilating each of them by itself, and iterating according to Equation 9, we see that Δ^* is both indefinitely long and thick. $\square$

In [10], Jean-Pierre Reveillès introduces a class of digital straight lines of Z^2 of variable thicknesses, by putting

$$D = \{(x, y) : t_0 \leq ax + by < t + t_0\} \qquad a, b \in Z, \tag{10}$$

where a and b are relatively prime. The term t_0 corresponds to a shift that can be taken equal to zero, and the term t to the thickness of D. In particular, when $t = \max(\mid a \mid, \mid b \mid)$, line D is said to be "naive". The extension to hyperplanes of Z^n is straightforward, one just has to replace $ax + by$ by $\sum a_i x_i$ in Equation 10. We draw from the equivalence

$$0 \leq \sum a_i x_i < t \quad \Leftrightarrow \quad \{\sum a_i x_i = s, \;\; 0 \leq s < t\}, \tag{11}$$

that the hyperplanes of Equation 10 represent, for every sub-module A of dimension $n - 1$, *space slices* $\Pi(A, c, c') = E(A, c') \backslash E(A, c)$, $c' \geq c$, which are parallel to A. For example, Figure 3 depicts the decomposition of a naive Reveillès line into a union of Bezout lines.

Definition 3 (Digital Steiner set)**.** A set $K \in \mathcal{K}(Z^n)$ is said to be *digital Steiner* if it can be decomposed into a finite Minkowski sum of Bezout segments centred at a same point. $\square$

Denote by $\mathcal{ST}(Z^n)$ the digital Steiner class. Theorem 1 implies that the Euclidean properties Equations 4 and (5) remain true in Z^n. Therefore every family of Steiner sets of increasing directional measures generates a granulometry by adjunction.

Clearly, the possible convexity of the Steiner sets did not play any role in the above analysis, and this point answers the first question set in introduction. We can however wonder how to link together convexity and digital Steiner sets, what we will do now.

5. Digital convexity and Steiner class

5.1 Digital convexity

The two definitions of convexity, by barycentre or by intersection of half-spaces, are no longer equivalent in Z^n as they were in R^n (see Figure 4(c)): we must choose between them. Indeed, the property of spanning of the space leads us to start from the second definition [12], p.171, [17], p.100-101.

Definition 4 (Digital Convexity)**.** A set $X \subseteq Z^n$ is said to be *convex* when it is equal to the intersection of all Bezout half-spaces that contain it, i.e., when

$$X = \cap\{E(A,c), c \in Z, \quad A \in \mathcal{A}\}$$

where $\mathcal{A}$ is the set of all sub-modules of dimension $n-1$ in Z^n. □

Given sub-module A, the smallest space slice $\Pi(A,c,c')$ parallel to A and which contains X is called the *supporting slice* $\Pi(X,A)$. We see that X is convex if and only if it is obtained by intersecting its supporting slices. This definition of the convexity implies the barycentre property since

Proposition 2. *Every Bezout segment in Z^n is convex. Moreover, if x, y are two points of a digital convex set X, then every point z of the Bezout segment $[x,y]$ belongs to X.*

Proof. We begin by the second part of the proposition. Let $\{H(\omega,c), c \in Z\}$ be a family of hyperplanes that span the space, and let c_x and c_y be the labels of the planes of two points x and y. The supporting slice $\Pi(\omega,x,y)$ generated by the hyperplanes $\{H(A,c), c \in [\ c_x,\ c_y]\}$ contains point z [15]. As set X is the intersection of its supporting half-spaces, and as for each A this intersection $C(X/A)$ contains the slice $\Pi(X,c_x,c_y)$, we can write

$$z \in \cap\{\Pi(X,c_x,c_y)\} \subseteq \cap\{C(X/A), A \in \mathcal{A}\} = X.$$

On the other hand, the intersection of all $\Pi(X,c_x,c_y)$ is nothing but the segment $[x,y]$. Indeed, suppose that a point $t \in \cap\Pi(X,c_x,c_y)$ does not belong to $[x,y]$. For any family $\{H(A,c)\}$ of hyperplanes $\{H(A,c)\}$ parallel to $[x,y]$, there exists a sense of ordering such that the labels c satisfy the conditions $c_x = c_y < c_t$, i.e., $[x,y] \in H(A,c_x)$. But $t \notin H(A,c_x)$, which implies that $t \notin \cap\Pi(X,c_x,c_y)$, i.e., that t must belong to segment $[x,y]$. □

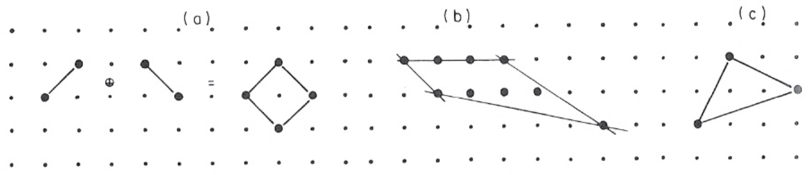


Figure 4. (a) This Steiner polygon is not convex, and, if we complete by the center of the cross, it becomes convex but is no longer Steiner; (b) convexity and connectivity are two independent notions; (c) for each pair (x, y) among the trois points, segment $[x, y]$ lies in the set, though the latter is not convex.

5.2 Convexity of the digital Steiner sets

Unlike in the Euclidean case, in Z^n the Minkowski sum of two segments of different directions is not necessarily convex (see Figure 4(a)). As the Steiner sets are built by means of such sums, the question arises to find under which conditions a directional measure $\{k_i\omega_i, 1 \leq i \leq p\}$ in Z^n corresponds to a *convex* Steiner set. We start from the following lemma:

Lemma 1. *The Minkowski dilate of Reveillès hyper-plane* $\Pi = \{0 \leq \sum a_i x_i < t\}$ *by Bezout vector* $u = \{u_i\}$ *is itself a Reveillès hyper-plane if and only if* $| \sum a_i u_i | \leq t + 1$. *Then it has for equation one of the two forms*

$$\Pi_u \cup \Pi = \{0 \leq \sum a_i x_i < \sum a_i u_i + t\} \text{ or } \{\sum a_i u_i \leq \sum a_i x_i < t\}. \quad (12)$$

Proof. The translate Π_u of Π has for equation $0 \leq \sum a_i(x_i - u_i) < t$, thus it is the hyper-plane $\sum a_i u_i \leq \sum a_i x_i < t + \sum a_i u_i$. The union $\Pi_u \cup \Pi$ is still an hyper-plane if and only if the two sequences of integers $[0, t]$ and $[\sum a_i u_i, t + \sum a_i u_i]$ are consecutive, i.e., when $\sum a_i u_i \leq t + 1$ or $\sum a_i u_i + t \geq -1$, or again when$| \sum a_i u_i | \leq t + 1$, which result in Equation 12. □

Proposition 3. *The Steiner set* $X \subseteq Z^n$ *of directional measure* $\{k_i\omega_i, 1 \leq i \leq p\}$ *is convex if and only if for all directions* ω_i *the dilate* $\Pi_i = X \oplus D_i$, *where* D_i *is the Bezout straight line supporting* L_i, *is an intersection of Reveillès hyperplanes, or again if the sequence of dilations that generates* X *is also a sequence of contiguous translations of* D_i.

The proof of Proposition 3 is given in [15].

Figure 5 illustrates the criterion by both an example and a counter-example. Take for L_p vector $(3, 2)$. In case (a), the translations of the line $2x - 3y = 0$ by vectors L_1 and L_2 of horizontal and vertical directions lead to the lines $2x - 3y = c$, with $c = \{-3, -1, 0, 1, 2, 4\}$, which is not a contiguous series, and also X is not convex. In case (b), segment L_3 à 45 has been added, which implies $-4 \leq c \leq 4$, and also that X becomes convex.

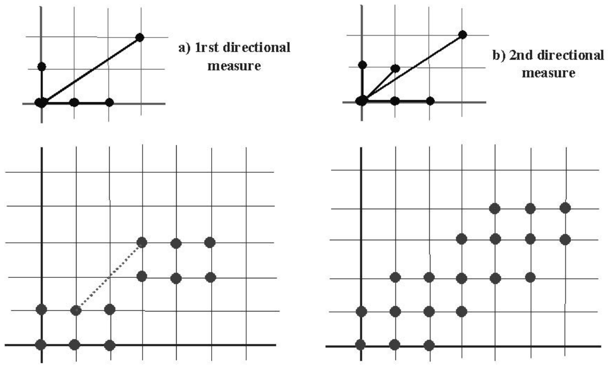


Figure 5. (a) Non convex but connected Steiner set. (b) Convex and connected Steiner set.

5.3 Connectivity of the digital Steiner sets

In Z^n, convexity does not imply connectivity (see Figure 4), even for convex Steiner sets (Figure 6(b)). However, in case of the usual arcwise connectivity, the conditions for the connectedness of a Steiner set can be found. With all point $x \in Z^n$ associate the unit cube $B(x)$ of centre x and whose points of the boundary define the extremities of the elementary arcs of origin x (e.g., the 8-connectivity in Z^2). As every cube $B(x), x \in Z^n$ contains the points which are just before and just after x in each direction of the axes, the parallelepipeds parallel to the axes are connected. Then we can state the following criterion

Proposition 4. *Let $\mathcal{C}$ be the arcwise connection on $\mathcal{P}(Z^n)$ generated by the unit cubes $B(x), x \in Z^n$. Consider a digital Steiner set $X \subseteq Z^n$ of directional measures $\{k_i\}$ in the directions $\{\omega_i\}$, $1 \leq i \leq p$, with $n \leq p$, and whose the n first directions are those of the axes of Z^n. The set X is then connected according to $\mathcal{C}$ if and only if for each j such that $n < j \leq p$ the component ω_i^j of direction ω_j w.r.t. to axis ω_i satisfies the inequality*

$$k_j \omega_i^j \leq k_i, \quad 1 \leq i \leq n+1, \qquad n < j \leq p. \tag{13}$$

Proof. Set X is written

$$X = L_1(k_1\omega_1) \oplus .. \oplus L_i(k_i\omega_i) .. \oplus L_p(k_p\omega_p) \tag{14}$$

where L_i is the vector of length k_i in direction ω_i. The dilate of origin O by the n vectors $\overrightarrow{k_i\omega_i}, 1 \leq i \leq n$, along the directions of the axes is a connected parallelepiped Π_0. Consider one of the supplementary directions $\omega_j, n < j \leq p$. The inequality in (13) means that the extremity z_j of the segment of length k_i in direction ω_j belongs to the dilate $\Pi_0 \oplus B$. As $O \in \Pi_0$, we have that $z_j \in \Pi_{z_j}$ therefore if $z_j \in \Pi_0$ then $z_j \in \Pi_0 \cap \Pi_{z_j}$ and the union

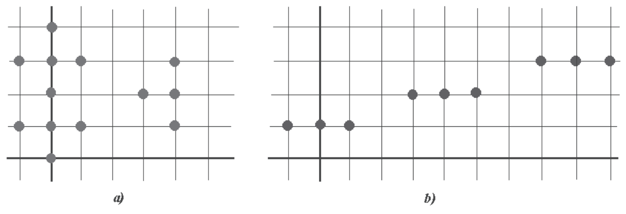


Figure 6. (a) The left set is open by the unit cross, but not by the right set, though smaller than the unit cross. (b) An example of non connected convex Steiner set.

$\Pi_0 \cup \Pi_{z_j}$ is connected. If $z_j \in (\Pi_0 \oplus B)\backslash\Pi_0$, then Π_0 and Π_{z_j} are adjacent, and their union is still connected. By iterating the proof for all directions $\omega_j, n < j \leq p$ we conclude that the Steiner set X is connected .

Conversely, suppose that X has several connected components, or "grains". As Π_0 is connected, it is included in one of the grains X_0 de X. Then Equation 14 implies that in one of the supplementary directions at least, j say, with $n < j \leq p$, the translate of the origin by vector $k_j\omega_j$ belongs to a grain of X disjoint form X_0, (if not, X would be a unique grain), hence disjoint from Π_0. Therefore, for label j, the inequalities (13) are not satisfied. $\square$

6. Perpective vision and structuring function

There are various ways to relax digital translation invariance. The one we develop in this section aims to describe the perspective mapping. We firstly observe that in Z^1 all Steiner openings are trivial on segments (i.e., suppress them or leave them unchanged), and we want to preserve this property under perspective changes. We shall proceed by reducing Z^1 by elementary removals. Start from an opening $\gamma = \delta\varepsilon$, of extensive primitive δ, and which is trivial on segments. Remove one arbitrary point from Z^1, taken as the origin, and and join together the two reduced half axes. The structuring elements $\{\delta(x), x \in Z^1\}$, once modified, generate a new function δ^*, still extensive and made of segments, according to the rules expressed in Figure 7, namely:

Location of the extremities of $\delta(x)$	$\delta(x)$	$\delta^*(x^*)$	x^*
$x_n \leq 0$	$\{x_1, ..., x_n\}$	$\{x_1, ..., x_n\}$	x
$x_1 \leq 0 < x_n$	$\{x_1, .., 0\}\{1, .., x_n\}$	$\{x_1, .., 0\}\{0, ..x_{n-1}\}$	0
$0 < x_1$	$\{x_1, ..., x_n\}$	$\{x_1 - 1, ..., x_n - 1\}$	$x - 1$

(15)

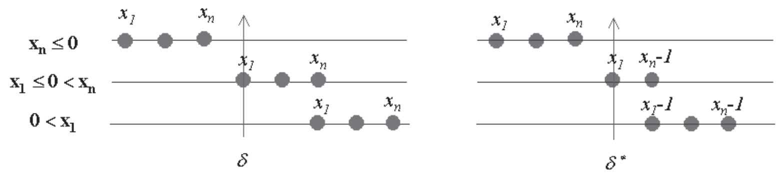


Figure 7. Bending of a structuring function δ by removal of the origin.

Consider now the action of opening $\gamma^* = \delta^*\varepsilon^*$ on a segment $L \subseteq Z^1$ of extremities y_1, y_p :

- when $y_p \leq 0$, then $\gamma^*(L) = \gamma(L)$,

- when $y_1 \leq 0 < y_p$, then, we have $\gamma^*(L) = [\gamma(L \oplus D)] \ominus D$, where D stands for doublet $\{0, 1\}$ centered at the origin,

- finally, when $0 < y_1$, then $\gamma^*(L) = \gamma(L \oplus \{1\})$.

In all cases, γ^* leaves L unchanged, or removes it, so that we can state the next proposition.

Proposition 5. *Let $\delta : Z^1 \to \mathcal{P}(Z^1)$ an extensive structuring function δ, with $\delta(x) \in \mathcal{ST}(Z^1)$, $x \in Z^1$. The structuring function δ^* defined by system (15) is in turn extensive, with $\delta^*(x) \in \mathcal{ST}(Z^1)$, $x \in Z^1$. Moreover, if the opening by adjunction $\gamma = \delta\varepsilon$ is trivial for segments, then $\gamma^* = \delta^*\varepsilon^*$is also trivial for segments.*

The construction can be iterated, and serves when the deformations are due to an oblique perspective (e.g., T.V camera watching an a road). The space Z^2 being indicated by two axes Ox and Oy (depth), one removes all the more parallel lines to Ox since their y-ordinates increase. At the same time, in the lines which are left, the translation invariant δ are reduced as y increases.

7. Conclusion

It was shown that, in Z^n, a Steiner class can be obtained uniquely when one starts from Bezout straight lines (i.e., the narrowest digital lines), and that the resulting sets generate granulometries but may be neither convex not connected. However these two properties are reached when the Steiner class satisfies Propositions 3 and 4. Most of the results proved for Z^n extend to the sphere and to the boundary of the discrete torus. The main new results of the developed approach are given by Theorem 1, Propositions 3 and 4, and by Propositions 2 and 5.

References

[1] U. M. Braga-Neto and J. Goutsias, *A Multiscale approach to Connectivity*, Computer Vision and Image Understanding **89** (2003), 70-107.

[2] U. Eckardt, *Digital lines and digital convexity*, Lecture Notes in Computer Sciences **2243** (2001), 209-228.

[3] H. Heijmans and C. Ronse, *The algebraic basis of mathematical morphology, I: erosions and dilations*, Computer Vision, Graphics and Image Processing **3** (1990), 245-295.

[4] R. Jones and P. Soille, *Perodic lines and their applications to granulometries*, Mathematical Morphology and its applications to image and signal processing, Maragos P. et al eds., Kluwer (1996), 263-272.

[5] C. Kiselman, *Convex functions on discrete sets*, In:IWCIA 2004. R. Klette and J. Zunic (Eds.)Lecture Notes in ComputerScience **3322** (2004), 443-457.

[6] G. Matheron, *Random sets and integral geometry*, Wiley, New-York, 1975.

[7] ______, *Les treillis compacts. Tech. rep.*, Ecole des Mines de Paris, Paris, 1996.

[8] E. Melin, *Digital straight lines in Khalimsky plane*, Uppsala Un. Dpt. of Mathematics, Report 2003:30, (accepted for publication in *Mathematica Scandinavia*), 2003.

[9] K. Murota, *Discrete Convex Analysis*, SIAM, Philadelphia, USA, 2003.

[10] J-P. Reveillès, *Géométrie discrète, calcul en nombres entiers et algorithmique*, Strasbourg: Université Louis Pasteur, Thèse d'Etat, 251 pp, 1991.

[11] A. Rosenfeld, *Digital straight lines segments*, IEEE transactions on Computers **c-32** (1974), no. 12, 1264-1269.

[12] J. Serra, *Image analysis and mathematical morphology*, Academic Press, London, 1982.

[13] ______, *Image analysis and mathematical morphology, part II: theoretical advances*, Academic Press, London, 1988.

[14] ______, *Set connections and discrete filtering*, G. Bertrand, M. Couprie and L Perroton (eds.), Lecture Notes in Computer Science, Springer **1568** (1999), 191-206.

[15] ______, *Convexes digitaux de Steiner et semi-groupes de Matheron, techn. report*, Ecole des Mines, Paris, 27p, 2007.

[16] P. Soille, E. J. Breen, and R. Jones, *Recursive Implementation of Erosions and Dilations Along Discrete Lines at Arbitrary Angles*, IEEE PAMI **18** (1996), no. 5, 562-567.

[17] P. Soille, *Morphological image analysis*, Springer-Verlag, Berlin Heidelberg, 1999.

[18] P. Soille and H. Talbot, *Directional Morphological Filtering*, IEEE PAMI **23** (2001), no. 11, 1313–1329.

[19] H. Talbot, *Analyse morphologique de fibres minérales d'isolation*, Ecole des Mines de Paris, Thèse de Doctorat en Morphologie Mathématique, 266 pp, 1993.

[20] H. Talbot and H. Heijmans, *The algebraic basis of mathematical morphology – Part II: Openings and closings.*, Computer Vision, Graphics and Image Processing :Image Understanding **54** (1991), 74-97.

The random spread model

Jean Serra

Laboratoire L2ASI, ESIEE, France
`serraj@esiee.fr`

Abstract The paper proposes the new stochastic model of a Random Spread for describing the spatial propagation of sequential events such as forest fires. A Random Spread is double Markov chain, whose each step is a (random) set operator β combining a Cox process with a Boolean random closed set. Under iteration, operator β provides the time evolution of the Random Spread, which turns out to be a birth-and-death process. Average sizes, and the probabilities of extinction are derived. The random spread model was applied to the analysis of the fires that occurred in the State of Selangor (Malaysia) from 2000 to 2004. It was able to predict all places where burnt scars actually occurred, which is a strongly significant verification.

Keywords: Random Spread, Boolean RACS, Markov chain, birth-and-death, random closed sets, forest fires, simulation, mathematical morphology.

1. Introduction

In a large number of wild forests, such as typically in South East Asia, forest fires propagate less under the action of the wind, as in Mediterrenan countries, than under almost isotropic causes. For instance, the fires that occurred in the state of Selangor (Malaysia, see Figure 1) from 2000 to 2004 seem to evolve randomly: the initial seats vanish sometimes, but may also give birth to new seats at some distance. The foresters use to describe the fire progress by means two key maps [16], namely the daily *spread rate* of the fire, i.e., the radius r of the daily circular propagation of the fire, and the *fuel consumption* f_w, (a weighted version of the vegetation map). Both are depicted in Figure 1. Clearly, a straightforward use of such key maps does lead to a pertinent description. By starting from *any* point seat, one always arrives to burn the whole country in a finite time, under iteration, as both maps are positive. Indeed, one must use the rate information in some restrictive way, to be able to reach actual events. It is exactly the purpose of the present paper.

2. Stochastic models for growth

One can classify the proposed model as a discrete branching process which generalises the Galton-Watson process, since it involves a location of the new

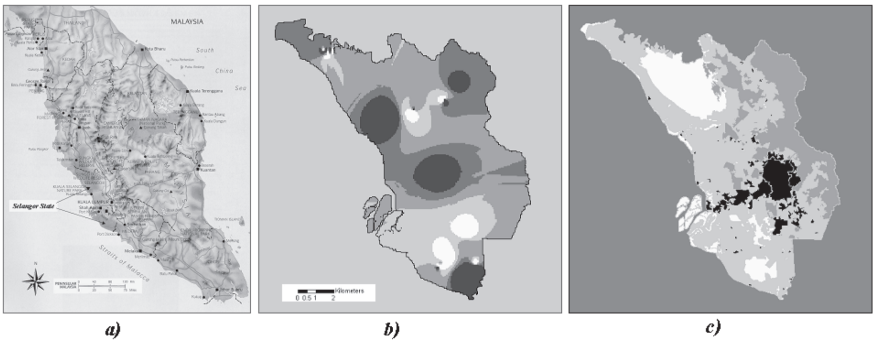


Figure 1. (a) State of Selangor in Malaysia. (b) Map of the spread rate, i.e., of the radius r of the daily circular propagation of the fire. (c) Map f_w of the fuel consumption.

generation in space. Since C. J. Preston's pioneer work on *spatial birth-and-death* [10], one finds in the literature several point stochastic processes for describing joint evolutions in space and time. Their characteristic functionals $Q(K)$ are generally inaccessible, but they yield significant simulations. A second class is the concern of "thick" structures, i.e., which do not reduce to isolated points. A particular case can be found in [11], p.562, under the name of hierarchical RACS, with several variants such as the following: *the RACS at time t is X_t, and X_{t+dt} is generated by adding to X_t any boolean grain which occurs during $[t, t+dt]$ and whose center hits X_t.* In spite of its outward simplicity, this variant is not tractable, i.e., one cannot express the functional Q_t of X_t by means of the initial conditions, as proved by D. Jeulin in [5].

However, a number of phenomena, including forest fires, follows the same type of behavior. Each time that in the mineral, vegetal or animal worlds, seeds move and then develop a new colony, they involve some random sequential growth. But how to model it by tractable random closed set (RACS)? The trouble with the hierarchical RACS comes from that their evolution between steps n and $n+1$ refers to the *whole past*, from 0 to n. If we relax this condition, can we reach more tractable growth RACS? In addition, we must take into account that the space parameters which govern the evolution laws (e.g. the fuel amount for forest fires) usually vary from place to place, so that the new model should not be *a priori* translation invariant, but accept some imposed heterogeneity.

That are the questions we consider in this paper, by proposing the *Random Spread* RACS.

3. Reminders

The random spread model in presented in the framework of the Euclidean space $\mathbb{R}^d$ of dimension d. Denote by the $\mathcal{P} = \mathcal{P}(\mathbb{R}^d)$ (resp. $\mathcal{F}$, $\mathcal{K}$) the family of all sets (resp. closed sets, compact sets) of $\mathbb{R}^d$. Symbol $\mathcal{S}$ stands for the singletons of $\mathcal{P}(\mathbb{R}^d)$ and the same symbol, e.g. x, is used for the points of $\mathbb{R}^d$ and for the elements of $\mathcal{S}$.

Set dilation A *structuring function* is an arbitrary mapping $x \longmapsto \delta(x)$ from $\mathcal{S}(\mathbb{R}^d)$ into $\mathcal{P}(\mathbb{R}^d)$. It extends into the *dilation* δ from $\mathcal{P}(\mathbb{R}^d)$ into itself [4,7,12] by the relation

$$\delta(X) = \bigcup\{\delta(x)\ ,\ x \in \mathcal{S}\} \qquad\qquad X \in \mathcal{P}(\mathbb{R}^d). \tag{1}$$

In the following, all structuring functions are supposed to be *compact*, i.e.,

Definition 1. A structuring function $\delta : \mathcal{S}(\mathbb{R}^d) \longmapsto \mathcal{P}(\mathbb{R}^d)$ is said to be *compact* when

i) it is upper semi continuous from $\mathcal{S}$ into $\mathcal{K}$,

ii) the union $\bigcup\{\delta_{-x}(x), x \in \mathbb{R}^d\}$ has a compact closure δ

$$\overline{\delta_0} = \overline{\bigcup\{\delta_{-x}(x), x \in \mathbb{R}^d\}}. \tag{2}$$

By extension, the associated dilation $\delta : \mathcal{P} \longmapsto \mathcal{P}$ is also said to be compact. □

The *reciprocal duality* between dilations plays an important role below. With each structuring function δ, associate the *reciprocal structuring function* ζ by writing

$$y \in \zeta(x) \qquad \text{if and only if} \qquad x \in \delta(y) \qquad\qquad x, y \in E. \tag{3}$$

The algorithm that expresses ζ in function of δ is therefore

$$\zeta(x) = \bigcup\{y : x \in \delta(y)\}. \tag{4}$$

Random Closed Sets (Euclidean case) The results which follow, on RACS and Boolean RACS, are basically due to G. Matheron [8]. Instructive introductions to RACS may be found in [9], or [15]. Given an element $K \in \mathcal{K}$, consider the class

$$\mathcal{F}^K = \{F : F \in \mathcal{F},\ F \cap K = \emptyset\}$$

of all closed sets that miss K.

As K spans the family $\mathcal{K}$, the classes $\{\mathcal{F}^K,\ K \in \mathcal{K}\}$ are sufficient to generate the σ-algebra. Moreover, as set $\mathcal{F}$ is compact for the hit-or-miss topology, there exist probabilities, Pr say, on σ_f, and each triplet $(\mathcal{F}, \sigma_f, \Pr)$ defines a RACS.

Theorem 1. *The distribution of a RACS X is* uniquely *defined by the datum of the probabilities*

$$Q(K) = \Pr\{K \subseteq X^c\} \tag{5}$$

as K spans the class $\mathcal{K}$ of the compact sets of $\mathbb{R}^d$. Conversely, a family $\{Q(K),\ K \in \mathcal{K}\}$ defines a (necessarily unique) RACS if and only if $1 - Q(K) = T(K)$ is an alternating Choquet capacity of infinite order such that $0 \leq T \leq 1$ and $T(\emptyset) = 0$.

The mapping $Q : \mathcal{K} \rightarrow [0,1]$ is called the characteristic functional *of the RACS.*

The characteristic functional Q plays w.r.t. a random closed set X the same role as the distribution function for a random variable x. If $\{X_i, i \in I\}$ stands for all possible realizations of the RACS X, and if $\psi : \mathcal{F} \longmapsto \mathcal{F}$ is semi-continuous, hence measurable, then the family $\{\psi(X_i),\ i \in I\}$ characterizes in turn all realizations of a RACS, denoted by $\psi(X)$. This basic result allows us to play with RACS just as with deterministic sets, to intersect them, to dilate them, etc.

Boolean RACS The Boolean RACS, is very popular and led to many variants (e.g., [1, 6, 11, 15]). In the present study, we specify it as follows. Consider the two primitives of

i/ a Poisson process $\mathcal{J}(\theta)$, whose intensity measure θ is upperbounded, i.e., $\theta(dx) \leq \overline{\theta}.dx$ with $\overline{\theta} < \infty$

ii/ a compact, and deterministic, structuring function $\delta : \mathcal{S}(\mathbb{R}^d) \rightarrow \mathcal{K}(\mathbb{R}^d)$ called “primary grain”.

The Boolean RACS X is constructed in two steps. First, take a realization J of Poisson points, which provides the set of points x_j, $x_j \in J$. Second, take the union X of all primary grains whose centers belong to the Poisson realization, i.e.,

$$X = \bigcup\{\delta_{x_j}, x_j \in J\}.$$

This union generates a realization of the Boolean RACS X. The characteristic functional Q of RACS X derives easily form the above definition [8], and equals

$$Q(K) = \exp - \int \theta(dz) 1_{\delta(z)\cap K} = \exp - \int \theta(dz) 1_{z\cap\zeta(K)} = \exp -\theta[\zeta(K)]. \tag{6}$$

where ζ is the dilation reciprocal of δ. If we restrict the previous Poisson points to those which occur in a given compact set X_0, then this comes back to change the intensity $\theta(dx)$ into $\theta^*(dx) = \theta(dx).1_{X_0}(x)$, the Boolean structure being preserved, so that

$$\Pr\{K \subseteq X_1^c\} = Q_1(K) = e^{-\theta^*[\zeta(K)]} = e^{-\theta[\zeta(K)\cap X_0]}. \tag{7}$$

4. Random Spreads

4.1 Definition

The Random Spread model generalizes Matheron's Boolean RACS by introducing a genetic dimension, namely the successive steps, according to which the $(n+1)^{th}$ Boolean RACS depends on the realization of the n^{th} one.

Consider an initial random seat I_0 made of an a.s. locally finite number of initial point seats in $\mathbb{R}^d$. The fire evolution from I_0 is the concern, on the one hand, of the fire the initial seats provoke, or *fire spread* $X_1 = \delta(I_0)$, and on the other hand of the generation of subsequent *seats spread* $I_1 = \beta(I_0)$. These secondary seats will develop new fires in turn. Both aspects refer to some compact dilation δ. We propose to model the seats spread $\beta(I_0)$ by picking out, randomly, a few points in each dilate $\delta(x_i)$, for all points $x_i \in I_0$. The double spread process is then written for the spread of

$$\text{the fire } X_1(I_0) = \delta(I_0) = \bigcup\{\delta(x_i),\ x_i \in I_0\}, \tag{8}$$

$$\text{the seats } I_1(I_0) = \beta(I_0) = \bigcup\{(\delta(x_i) \cap J_i)\,,\ x_i \in I_0,\ J_i \in \mathcal{J}(\theta)\}, \tag{9}$$

where a different realization J_i of the Poisson points process $\mathcal{J}(\theta)$ is associated with each point x_i. Therefore, each point x_i of the set I_0 induces a bunch of seats $\delta(x_i) \cap J_i$ independent of the others. These two equations mean that though the fire from a seat x does burn the zone $\delta(x)$ around x, only a few points of the scar $\delta(x)$ remain active seats for the next step.

Under iteration, Equations 8 and 9 become

$$X_2(I_0) = \delta(I_1) = \bigcup\{\delta(y_k),\ y_k \in I_1\} = \bigcup\{\delta(y_k),\ y_k \in \delta(x_i) \cap J_i\ ;\ x_i \in I_0\},$$

$$I_2(I_0) = \beta(I_1) = \beta[\beta(I_0)] = \bigcup_i\{\bigcup_k[\delta\,(\delta(x_i) \cap J_i)] \cap J_k\},\ x_i \in I_0\}.$$

Figure 2 depicts the first three steps of a random spread, for which:

- the initial seat I_0 is the point x_0, and the first spread, or front, the dark grey disk $X_1(I_0) = \delta(x_0)$;
- then two Poisson points, namely y_1 and y_2, fall in $\delta(x_0)$. They generate

  $X_2(x_0) = \delta(y_1) \cup \delta(y_2) = \delta(I_1)$, in medium grey,
  and $I_1(x_0) = \{y_1\} \cup \{y_2\}$;

- then again a new Poisson realization generates one point, z_1 in $\delta(y_1)$, and another Poisson realization the three points $z_{2,1}, z_{2,2}$, and $z_{2,3}$ in $\delta(y_2)$, hence

  $X_3(x_0) = \delta(z_1) \cup [\delta(z_{2,1}) \cup \delta(z_{2,2}) \cup \delta(z_{2,3})] = \delta(I_2)$, in light grey,
  and $I_2(x_0) = \{z_1\} \cup [\{z_{2,1}\} \cup \{z_{2,2}\} \cup \{z_{2,3}\}]$.

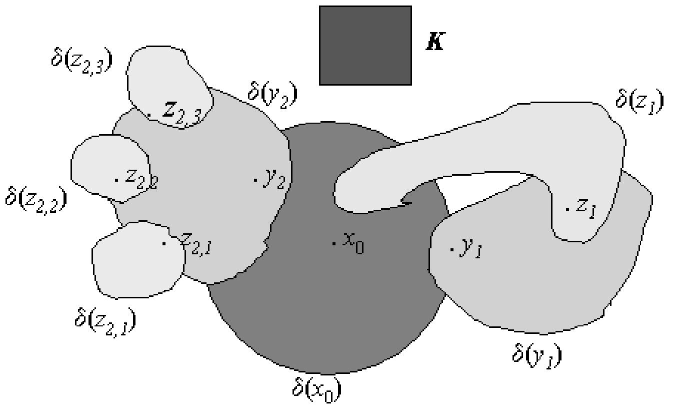


Figure 2. Three generations of fires stemming from point $x_0 = I_0$. Note the generation of new fires in already burnt areas.

The doublet spread (X_n, I_n) of order n, is the n^{th} element of the chain depicted in Figure 3, which is of Markov type: as soon as I_n is known, the previous links do not serve in the creation of link (X_{n+1}, I_{n+1}). This Markov type assumption means that the fire of tomorrow can only be caused by points seats stemming from the zone which burns today. What burnt yesterday, before yesterday, etc., has no longer importance. In this space-time process, the successive sets $X_1, X_2 ... X_n$ are at least as descriptive as the seats $I_1, I_2 ... I_n$ themselves, because they show the extension of the fire front through the time steps. The $(n+1)$ step is obtained by the two induction relations

$$X_{n+1}(I_0) = \delta[I_n(I_0)] \tag{10}$$

$$I_{n+1}(I_0) = \beta[I_n(I_0)]$$

$$I_{n+1}(I_0) = \bigcup\{(\delta(x_{i,n}) \cap J_{i,n}), \quad x_{i,n} \in I_n, \quad J_{i,n} \in \mathcal{J}(\theta)\}. \tag{11}$$

We observe that, in Equation 9, and the ulterior ones, each small zone dz intervenes as many times as dz belongs to different $\delta(x_i)$. Therefore, the measure

$$\tau_{n+1}(dz) = \theta(dz) \sum \{1_{\delta(x_{i,n})}(z), \quad x_{i,n} \in I_n\} \tag{12}$$

turns out to be a realization of the Cox process of intensity τ_{n+1}, if I_{n+1} is finite. Now, are we sure that all these handlings and derivations really lead to a RACS, in Matheron-Kendall's sense, i.e., to something we can characterize by a functional $Q(K)$? Are we sure, for example, that the intensity τ_{n+1} is always a.s.finite? The answer is positive, as soon as the structuring function δ is *compact*, in the sense of Definition 1. Then one can state the following proposition [13].

$$\begin{tikzcd}
 & X_1=\delta(I_0) & X_2=\delta(I_1) & & X_{n+1}=\delta(I_n) \\
I_0 \arrow[r] \arrow[ur] & I_1=\beta(I_0) \arrow[r] \arrow[ur] & I_2=\beta(I_1) \arrow[r, phantom, "\ldots\ldots"] & I_n=\beta(I_{n-1}) \arrow[r] \arrow[ur] & I_{n+1}=\beta(I_n)
\end{tikzcd}$$

Figure 3. The Markov chain of the spreads.

Proposition 1. *Let I_0 be a a.s. locally finite set of points, let θ be a Poisson intensity, and $\delta : \mathbb{R}^d \to \mathcal{K}(\mathbb{R}^d)$ be a structuring function. If function δ is compact, then both families $\{X_n(I_0), n > 0\}$ and $\{I_n(I_0), n > 0\}$ of fire spreads and seats spreads are RACS.*

Since the two spreads I_n and X_n are RACS, we shall characterize them by their functionals $Q_n(K)$. In the "Boolean-Cox RACS" X_n, the primary grain *only* is independent of step n, since at each time n, the intensity τ_n is a new one. This circumstance simplifies the theoretical study of the time evolution. Also, it suggests to find induction relations between $Q_n(K)$ and $Q_{n+1}(K)$ that reflect the two definitions by induction of Equations 10 and 11.

4.2 Characteristic functional

The additivity property of the random spread allows us to take for I_0 a point initial seat, x_0 say, of dilate $X_1 = \delta(x_0)$, and whose intersection of the dilate with Poisson points J provides the first random set $I_1 = \delta(x_0) \cap J$. The functional $Q_n(K \mid x_0)$ of the random fire spread $X_n(x_0)$, i.e., the probability that set K misses the n^{th} spread $X_n(x_0)$ of initial seat x_0, satisfies an induction relation between steps n and $n+1$. The compact set K lies in the pores of the $(n+1)^{th}$ spread if and only if none of the points $y \in \delta(x_0)$ can develop a n^{th} spread that hits K. For a given $y \in \delta(x_0)$, this elementary probability is

$$dQ_{n+1}(K \mid x_0 \mid y) = 1 - \theta(dy) + \theta(dy)Q_n(K \mid y), \qquad dy \in \delta(x_0). \tag{13}$$

As the events occurring in disjoint dy are independent, we obtain $Q_{n+1}(K \mid x_0)$ by taking the infinite product inside $\delta(x_0)$, i.e.,

$$Q_{n+1}(K \mid x_0) = \exp - \int_{\delta(x_0)} \theta(dy)[1 - Q_n(K \mid y)]. \tag{14}$$

Each step involves an exponentiation more than the previous one. We find for example for the first steps that

$$Q_2(K) = \exp -\theta[\zeta(K) \cap \delta(x_0)], \tag{15}$$
$$Q_3(K) = \exp - \int_{\delta(x_0)} \theta(dy)[1 - e^{-\theta[\zeta(K)\cap\delta(y)]}],$$
$$Q_4(K) = \exp - \int_{\delta(x_0)} \theta(dy)[1 - \exp\{- \int_{\delta(y)} \theta(dz)[1 - e^{-\int_{\delta(z)} \theta(dw) 1_{\zeta(K)}(w)}]\}].$$

where Q_2, but neither Q_3 nor Q_4, is equivalent to the Boolean RACS functional of Equation 7. The seat spread I_{n+1} satisfies the same induction relation 14 as the fire spread X_{n+1}. The only change holds on the first term, for which it suffices to replace $\zeta(K)$ by K in Equation 15. We can summarize the main results on the characteristic functional by stating:

Theorem 2. *Let*

- β be the random spread of parameters (θ, δ),

- $I_1 = \beta(x_0)$ be the random seat spread stemming from point x_0 of dilate $X_1 = \delta(x_0)$,

- $I_2 = \beta(I_1)$ and $X_2 = \delta(I_1)$ be the iterated seat spread and its fire spread,

- $I_{n+1} = \beta(I_n)$ be n^{th} iteration of β, and $X_{n+1} = \delta(I_n)$ the associated fire spread,

then the characteristic functionals of both RACS I_{n+1} and X_{n+1} satisfy the induction relation

$$Q_{n+1}(K \mid x_0) = \exp - \int_{\delta(x_0)} \theta(dy)[1 - Q_n(K \mid y)],$$

with initial terms

$$\begin{aligned} Q_1(K) &= \exp -\theta[K \cap \delta(x_0)] \text{ for the seat spread } I_1 \text{ and} \\ Q_2(K) &= \exp -\theta[\zeta(K) \cap \delta(x_0)] \text{ for the fire spread } X_2. \end{aligned}$$

4.3 Discussion

A Markov type assumption of order one underlies the random spread model, since it suffices to know the n^{th}seats for constructing the $(n+1)^{th}$ ones. This assumption just allowed us to achieve the formal calculus of the functionals Q_n and R_n. But it does not prevent us from the comeback of new seats on already burnt areas, after a few steps. The successive spreads may turn around $\delta(x_0)$, as depicted in Figure 2.

The trouble can partly be overcome by a Markov assumption of order two, i.e., by making depend the $(n+2)^{th}$ seats of both the $(n+1)^{th}$ and n^{th} steps. We can impose for example to keep the second seats if they fall in the $\delta(x_i), x_i \in I_1$ but *not* in $\delta(x_0)$, and so on. The new version of the induction relation 14 must now involve two successive terms. Given the two points y and z the equation 13 of the elementary functional becomes

$$dQ_{n+1}(K \mid x_0 \mid y \mid z) = 1-\theta(dz)1_{\delta(y)\setminus\delta(x_0)}(z)+\theta(dz)1_{\delta(y)\setminus\delta(x_0)}(z)Q_n(K \mid z),$$

which gives, by integration in z

$$Q_{n+1}(K \mid x_0 \mid y) = \exp\left\{-\int_{\delta(y)\setminus\delta(x_0)} \theta(dz)[1-Q_n(K \mid z)]\right\}$$

and finally

$$Q_{n+2}(K \mid x_0) = \exp\left\{-\int_{\delta(x_0)} \theta(dy)[1-Q_{n+1}(K \mid x_0 \mid y)]\right\}$$

$$= \exp\left\{-\int_{\delta(x_0)} \theta(dy)[1-\exp\{-\int_{\delta(y)\setminus\delta(x_0)} \theta(dz)[1-Q_n(K \mid z)]\}]\right\}.$$

More generally, if the random spread of order n satisfies a Markov assumption of order n, one then obtains a new representation of the hierarchical RACS ([11], p.562, [5], Ch. 6), probably more tractable.

4.4 Spontaneous extinction

The fire which stems from the point seat x_0 may go out, spontaneously, after one, two, or more steps. The description of this phenomenon does not involve any particular compact set K. Denote by $g(n \mid x_0)$ the probability that the fire extinguishes after step n. This event occurs after the first step when no Poisson point falls inside set $\delta(x_0)$, i.e., with the probability

$$g(1 \mid x_0) = 1-\exp -\theta[\delta(x_0)].$$

The proof by induction that allowed us to link Q_{n+1} with Q_n in Equation 14 applies again, and gives, for a spontaneous extinction after step $n+1$, the probability

$$g(n+1 \mid x_0) = 1-\exp\{-\int_{\delta(x_0)} \theta(dy)g(n \mid y)\}. \tag{16}$$

For example, the probability $g(3 \mid x_0)$ of an extinction after the third step is:

$$1-\exp -\int_{\delta(x_0)} \theta(dy)[1-\exp\{-\int_{\delta(y)} \theta(dz)[1-\exp\{-\int_{\delta(z)} \theta(dw)\}]\}].$$

Suppose for the moment that θ and δ are translation invariant. The extinction probabilities no longer depend on x_0, y, etc., and reduce to $g(n+$

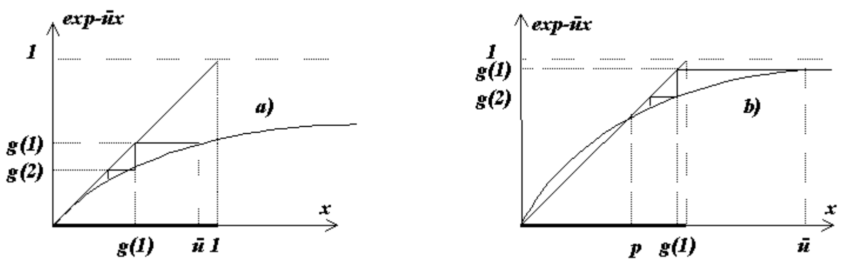


Figure 4. (a) The weight $\Vert u < 1$, then $g(n) \to 0$, as $n \to \infty$. (b) The weight $\Vert u \geq 1$, then $g(n) \to p$, as $n \to \infty$.

$1 \mid x_0) = g(n+1)$, $g(n \mid y) = g(n)$, etc. Similarly, the integral $\Vert u = \int_{\delta(z)} \theta(dx)$ is independent of $z \in \mathbb{R}^d$, so that

$$g(n+1) = 1 - \exp - \Vert u g(n). \tag{17}$$

As $n \to \infty$, the behavior of g depends on $\Vert u$. If $\Vert u < 1$, which corresponds to Figure 4(a), then $g(n) \to 0$, i.e., the fire extinguishes spontaneously, almost surely, in a finite time. When $\Vert u \geq 1$, then the two curves of Figure 4(b) intersect at point (p, p), where $p > 0$ is solution of the equation $p = 1 - \exp \Vert u p$. There is a non zero probability, namely p, of an infinite spread.

Suppose now that both functions θ and δ vary over the space, and let Z be the set of all points where $u(z) \geq 1$. If $x_0 \in Z$, then there is every chance that the fire invades the connected component of $\delta(Z)$ that contains point x_0. Such typical behaviour will be used now for predicting the scars of forest fires in the State of Selangor.

5. Scar prediction

The forestry services call *scar* of the fire the cumulative process

$$Y(x_0) = \bigcup \{X_p(x_0), 1 < p < \infty\}. \tag{18}$$

We will now match the data of the actual scars with the model. Section 4.4 has shown the crucial role played by the weight $u\,(x)$. In each region Z where all $u\,(x)$, $x \in Z$, are noticeably ≥ 1, any initial seat invades progressively the whole region, whereas in the regions with $u\,(x) < 1$, the spread stops by itself, all the sooner since $u\,(x)$ is small. In Selangor's case, the expression of u from the two maps of Figure 1 is as follows

$$u\,(x) = \int \theta\,(dz)\, 1_{\zeta(x)}\,(z) = k \int f_w\,(z)\, 1_{\zeta(x)}\,(z)\, dz \simeq \pi k f_w\,(x)\, r^2\,(x)\,.$$

This expression suggests to introduce the *scar function*

$$s(x) = f_w\,(x)\,r^2\,(x)\,,$$

as represented in Figure 5. Scar function s is accessible from the experimental data, since functions f_w and r are given (see Figures 1(c) and 1(a)). Note that map s is not obtained by simulations, but comes from a combination of the input parameters of the random spread model. By putting a threshold on image s at level $1/\ \pi k$, one splits the plane into the two regions where, either fires spontaneously extinguish (when $s(x) < 1/\ \pi k$), or invade the connected components that contain their initial seats (when $s(x) \geq 1/\ \pi k$).

If we take for k the value 1.61×10^{-3}, which derives from the hot spots measurements [16], we get $1/\ \pi k = 193$. The two sets above thresholds 190 and 200 are reported in Figure 5(b), side by side with the burnt areas (Figure 5(c)). In Figure 5(b), the fire locations **A** to **E** predicted by the model point out regions of actual burnt scares. Such a remarkable result could not be obtained from the maps f_w and r taken separately: the scare function $s = f_w r^2$ means something more, which corroborates the random spread assumption. Region **F** is the only one which seems to invalidate the model. As a matter of fact this zone is occupied by peat swamp forest, or rather, *was occupied.* It is today the subject of a fast urbanization, linking the international airport of Kuala Lumpur to the administrative city of Putra Jaya. Finally, on the whole, the random spread model turns out to be realistic.

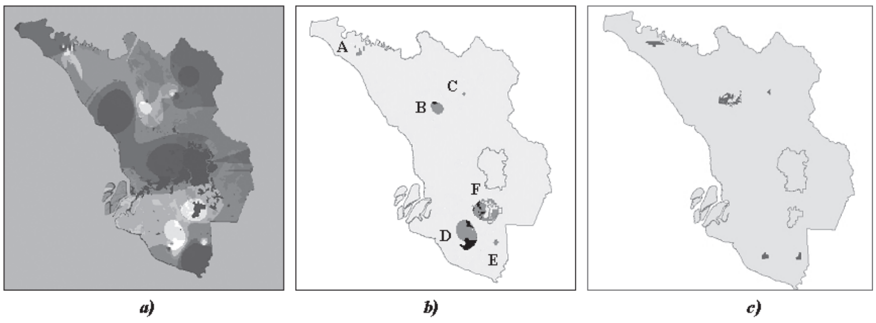


Figure 5. (a) Scar function $s = f_w \times r^2 = u/\pi k$ whose thresholds estimate the burnt scar zones. (b) Two thresholds of function s for $1/\pi k = 190$, in dark grey, and $1/\pi k = 200$, in black (the simulations suggest value 193). (c) Map of the actual burnt areas. Note the similarity of the sets, and of their locations.

6. Conclusion

This paper proposes a new RACS model, the *random spread*, which combines the three theoretical lines of Boolean random sets, Markov chains, and birth-and-death processes. Its characteristic functional was established. More than classical spatial birth-and-death processes, spread RACS depends strongly on the heterogeneity of the space, which appears via two functions of intensity (θ) and extension (δ). As a result, the process no longer describes a global birth-and-death, but regional expansions and shrinkages of the sets under study, namely the front, the seats and the scar of the spread RACS.

The time evolution was introduced in a discrete manner, by the Markov assumption that the fire front of tomorrow can only be caused by points seats stemming from the zone which burns today.

We draw from the model and a precise predictor of scars that actually occurred in the State of Selangor during the period 2000–2004.

References

[1] G. Ayala, J. Ferrandiz, and F. Montes, *Methods of Estimation In Boolean Models*, Acta Stereologica **8-2** (1989), 629–634.

[2] M. Bilodeau, F. Meyer, and M. Schmitt, *Space, Structure, and Randomness*, Lecture Notes in Statistics, Springer, Berlin, 2005.

[3] V. Cox and Isham, *Point Processes*, Chapmann and Hall, New-York, 1980.

[4] H. Heijmans and C. Ronse, *The algebraic basis of mathematical morphology, I: erosions and dilations*, Computer Vision, Graphics and Image Processing **3** (1990), 245–295.

[5] D. Jeulin, *Modèles morphologiques de structures aléatoires et de changement d'échelle*, Université de Caen, Thèse Doctorat ès Sciences Physiques, 410 p., 1991.

[6] ———, *Random texture models for materials structures*, Statistics and Computing **10** (2000), 121–131.

[7] C. Kiselman, *Digital Geometry and Mathematical Morphology*, Uppsala University, Lecture Notes 85p., 2002.

[8] G. Matheron, *Random sets and integral geometry*, Wiley, New-York, 1975.

[9] I. Molchanov, *Random closed sets*, In [2], 135–149.

[10] C. J. Preston, *Spatial Birth-and-death process*, Bull.Int. Stat. Inst. **46** (1977), 371–391.

[11] J. Serra, *Image analysis and mathematical morphology*, Academic Press, London, 1982.

[12] ———, *Image analysis and mathematical morphology, Volume II: theoretical advances*, Academic Press, London, 1988.

[13] ———, *Random Spread Tech. report*, Ecole des Mines de Paris, Paris 30 p., 2007.

[14] D. Stoyan, W. S. Kendall, and J. Mecke, *Stochastic Geometry and its Applications, 2*, Wiley, Chichester, 1995.

[15] D. Stoyan and J. Mecke, *The Boolean Model: From Matheron till today*, In [2], 151–181.

[16] M. D. H. Suliman, J. Serra, and M. A. Awang, *Morphological Random Simulations of Malaysian Forest Fires,in DMAI'2005*, AIT Bangkok, Chen X. Ed., 2005.

III

SIGNAL PROCESSING

A scale-space toggle operator for morphological segmentation

LEYZA B. DORINI and NEUCIMAR J. LEITE

Instituto de Computação (IC), Universidade Estadual de Campinas (Unicamp), SP, Brazil
{ldorini,neucimar}@ic.unicamp.br

Abstract The analysis of different representation levels has been largely used to handle the multiscale nature of image data. Here, we explore the scale-space properties of a toggle operator defined on a scaled morphological framework. These properties conduce to a well-controlled image extrema simplification, yielding sound segmentation and filtering results even when the operator is used in a binarization process. Also, the watershed transform using markers extracted from the processed image gives better segmentation results than when using markers from the original one. To show the robustness of our approach, we carried out tests on images of different classes and subjected to different lighting conditions.

Keywords: scale-space, scale-dependent morphology, image segmentation.

1. Introduction

Multiscale approaches have been largely considered in several signal processing applications, allowing the analysis of different representation levels and, further, the choice of the ones exhibiting the interest features.

One of the basic problems that arises when using multiscale methods originates from the difficulty to relate meaningful information of the signal across scales. In [20], Witkin proposed a novel multiscale approach, named scale-space, where the representation of an interest signal feature describes a continuous path through the scales. In such a way, it is possible to relate information obtained in different representation levels, as well as to have a precise localization of the interest features in the original signal.

Another important characteristic is that the transformation to a coarser level in the scale-space representation does not introduce artifacts, that is, signal features present at a scale σ are also present at all finer scales. This property is called monotonicity, since the number of features must necessarily be a monotonic decreasing function of scale [20].

Since the introduction of the scale-space theory, a large number of formulations have been proposed, based on different assumptions. In the linear approach, formalized by Witkin [20], a family of images is generated by convolving the original image with a Gaussian kernel. The signal extrema and

G. Banon, et al. (eds),
Mathematical Morphology and its Applications to Signal and Image Processing, 101–112.
©2007 *MCT/INPE. Printed in Brazil.*
URL of this article: http://urlib.net/dpi.inpe.br/ismm@80/2007/03.19.15.04

its first derivative constitute the features of interest. However, any convolution kernel used to obtain the scale-space introduces new extrema (the image maxima and minima) as the scale increases and, thus, the monotonicity property for linear filters and signal extrema does not hold [9]. To avoid this problem, other linear and non-linear approaches have been introduced [2].

Here, we explore the scale-space properties of a toggle-based operator in segmentation applications. The operator is defined in a scaled morphological framework using concave structuring functions, which was proved to have contrast properties [14]. Furthermore, it considers local pixel information (not only scale knowledge) to determine if each pixel should be processed by erosion or dilation, in contrast to other multiscale approaches that take into account mainly global information. All this characteristics conduce to an image simplification that enables the identification of important image structures using very simple operations, even in ill-illuminated images.

The next section presents basic multiscale morphology definitions. Section 3 defines the scale-space toggle operator and some of its main properties. Finally, we show some results in Section 4 and draw some conclusions and future work perspectives in Section 5.

2. Morphological-based scale-space

The notion of scale is related to the way we observe the physical world, where different features are made explicit at different scales. In mathematical morphology, the concept of scale (or size) dependent observations was introduced by Matheron [11] in his work on granulometry, which captures the size distribution of spatial observations.

To introduce the notion of scale, we can make the basic morphological operations of erosion and dilation scale-dependent by defining a scaled structuring function $g_\sigma : \mathcal{G}_\sigma \subset \mathbb{R}^2 \rightarrow \mathbb{R}$, such that [4]

$$g_\sigma(\mathbf{x}) = |\sigma| g(|\sigma|^{-1}\mathbf{x}) \quad \mathbf{x} \in \mathcal{G}_\sigma, \forall \sigma \neq 0, \tag{1}$$

where $\mathcal{G}_\sigma = \{\mathbf{x} : \|\mathbf{x}\| < \mathcal{R}\}$ is the support region of the function g_σ. To ensure reasonable scaling behavior, some other conditions are necessary [4], requiring a monotonic decreasing structuring function along any radial direction from the origin. In this paper, we use as structuring function $g(x,y) = -\max\{x^2, y^2\}$, that in the scaled version is given by

$$g_\sigma(x,y) = -|\sigma|^{-1} \max\{x^2, y^2\}, \tag{2}$$

where σ represents the scale. Observe that, for a 3×3 structuring element (used in this work), g_σ is zero at position $\mathbf{0}$ and $-|\sigma|^{-1}$ otherwise. Figure 1 illustrates the structuring function.

Scaled morphological operators have been frequently associated to nonlinear filters and scale-space theory. Jackway [4] introduced a scale-space

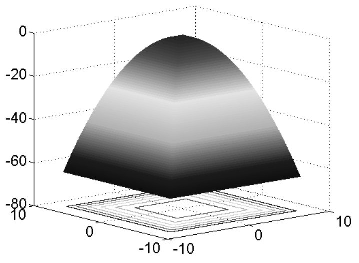


Figure 1. The structuring function.

based on the Multiscale Morphological Dilation Erosion (MMDE) operator, which unifies the scaled erosion and dilation transformations so that both positive and negative scales are taken into account (the image is processed by dilation, for positive scales, and by erosion for the negative ones). The interest features are the watershed of the smoothed signal in a certain scale. However, this method cannot be directly associated with image segmentation since "the watershed arcs moves spatially with varying scale" [3, 4].

In [7, 8], Leite and Teixeira explored the extrema preservation property of MMDE by using the extrema set obtained during the filtering process as markers in a homotopic modification of the original image, avoiding the spatial shifting of the watershed lines. They controlled the extrema merging through the different scales, obtaining good segmentation results. The authors also defined a new operator that explores the idempotence of the MMDE, establishing a relation between the structuring function g_σ and the extremes that persist at a given scale σ.

Scaled morphological operators have also been applied for image sharpening. Kramer [5] proposed a non-linear operator that replaces the original gray value of a pixel by the local minimum or maximum, depending on what value is closer to the original one. Shavemaker et al. [14] generalized this result by defining a new class of iterative scaled morphological image operators. In fact, they proved that all the operators that use a concave structuring function have interesting sharpening properties. Here, we explore some important characteristics of these operators to introduce a toggle transformation having interesting scale-space extrema preservation properties, as explained next.

3. Operator definition

A toggle operator has two major points: the primitives and a given decision rule [16]. Here, we use as primitives an extensive and an anti-extensive transformation, namely, the scale dependent dilation and erosion. The decision rule involves, at a point $\mathbf{x}$, the value $f(\mathbf{x})$ and the primitives results.

Formally:

$$(f \oslash g_\sigma)^k(\mathbf{x}) = \begin{cases} \psi_1^k(\mathbf{x}) & \text{if } \psi_1^k(\mathbf{x}) - f(\mathbf{x}) < f(\mathbf{x}) - \psi_2^k(\mathbf{x}), \\ f(\mathbf{x}) & \text{if } \psi_1^k(\mathbf{x}) - f(\mathbf{x}) = f(\mathbf{x}) - \psi_2^k(\mathbf{x}), \\ \psi_2^k(\mathbf{x}) & \text{otherwise,} \end{cases} \tag{3}$$

where $\psi_1^k = (f \oplus g_\sigma)^k$, that is, the dilation of f with the scaled structuring function g_σ, k times. In the same way, $\psi_2^k = (f \ominus g_\sigma)^k$.

In the following, we analyze the operator's behavior regarding on scale changing and on the recursive application of the primitives.

3.1 Changing the number of iterations

To avoid undesirable effects such as halos and oscillations, idempotent toggle operators are used [15]. Since the defined operator (Equation 3) is not idempotent, we use an alternative solution to this problem based on the specific knowledge of the pixels transformation. The following proposition formalizes an important property of the operator (see proof in [6]), which guarantees that it has a well-controlled behavior.

Proposition 1. *Let $\boldsymbol{x}$ be a pixel of the image and g be a structuring function with a single maximum at the origin, that is, $g(\boldsymbol{x})$ is a local maximum implies $\boldsymbol{x} = \boldsymbol{0}$. The sequence defined by $(f \oslash g_\sigma)^k(\boldsymbol{x})$ is stationary and monotonic increasing (decreasing) until a certain iteration k_0, while it is monotonic decreasing (increasing) after the iteration k_0.*

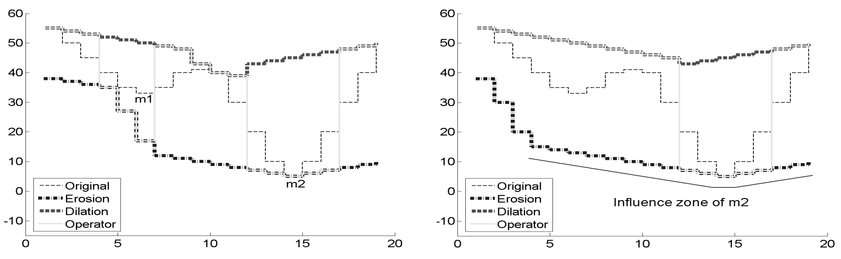
This property states that a pixel can initially converge to a specific local minimum and, after a certain iteration, to converge to an image maximum, or vice-versa [6]. Furthermore, since the sequence is stationary, we have the guarantee that it converges to a constant value, that is, it stabilizes after a certain number of iterations. Note that k_0 can be 1, that is, a pixel transformed value is strictly increasing or decreasing.

In Figure 2, as the number of iterations increases, the influence zone of the deeper minimum m_2 grows significantly, in such a way that the value of $f(m_1)$ become closer to the dilated values, and stabilizes after 10 iterations.

At this step, we can conclude that, in some neighborhood of an important minimum (maximum), the pixels values will be eroded (dilated) in such a way that it is possible to identify the significant extrema of the image and their influence zones. In this sense, we can define a new thresholding operation that uses a decision rule similar of that in Equation 3:

$$(f \oslash g_\sigma)^k(\mathbf{x}) = \begin{cases} 255 & \text{if } \psi_1^k(\mathbf{x}) - f(\mathbf{x}) <= f(\mathbf{x}) - \psi_2^k(\mathbf{x}) \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

where, again, $\psi_1^k(\mathbf{x}) = (f \oplus g_\sigma)^k(\mathbf{x})$, that is, the dilation of $f(\mathbf{x})$ with the scaled structuring function g_σ k times. In the same way, $\psi_2^k(\mathbf{x}) =$



(a) After seven iterations (b) After ten iterations

Figure 2. Operator behavior through different iterations using $g = [-1\ 0\ \ -1]$.

$(f \ominus g_\sigma)^k(\mathbf{x})$. This binary approach yields sound segmentation and filtering results, as we shall illustrate in Section 4.1.

3.2 Changing the scale

By taking into account the increasing of scale, the defined transformation (Equation 3) results in a new scale-space definition. The following proposition states that if an extrema of the signal is present at a given scale σ, it must be found at all the intermediate scales.

Proposition 2 (Behaviour of image extrema). *Let g be a structuring function with a single maximum at the origin, that is, $g(\boldsymbol{x})$ is a local maximum implies $\boldsymbol{x} = \boldsymbol{0}$. To avoid level-shifting and horizontal translation effects, we require that* $\sup_{t \in \mathcal{G}}\{g(\boldsymbol{t})\} = 0$, *and* $g(\boldsymbol{0}) = 0$. *Thus*

1. *if $(f \oslash g_\sigma)(\boldsymbol{x}_{max})$ is a local maximum, then $f(\boldsymbol{x}_{max})$ is a local maximum of $f(\boldsymbol{x})$ and $(f \oslash g)(\boldsymbol{x}_{max}) = (f \oplus g)(\boldsymbol{x}_{max}) = f(\boldsymbol{x}_{max})$,*

2. *if $(f \oslash g_\sigma)(\boldsymbol{x}_{min})$ is a local minimum, then $f(\boldsymbol{x}_{min})$ is a local minimum of $f(\boldsymbol{x})$ and $(f \oslash g)(\boldsymbol{x}_{min}) = (f \ominus g)(\boldsymbol{x}_{max}) = f(\boldsymbol{x}_{min})$,*

3. *If $0 < \sigma_1 < \sigma_2$ and $(f \oslash g_{\sigma_2})(\boldsymbol{x}_{max})$ is a local maximum, then $(f \oslash g_{\sigma_1})(\boldsymbol{x}_{max})$ is a local maximum and $(f \oslash g_{\sigma_1})(\boldsymbol{x}_{max}) = (f \oslash g_{\sigma_2})(\boldsymbol{x}_{max})$,*

4. *If $0 < \sigma_1 < \sigma_2$ and $(f \oslash g_{\sigma_2})(\boldsymbol{x}_{min})$ is a local minimum, then $(f \oslash g_{\sigma_1})(\boldsymbol{x}_{min})$ is a local minimum and $(f \oslash g_{\sigma_1})(\boldsymbol{x}_{min}) = (f \oslash g_{\sigma_2})(\boldsymbol{x}_{min})$.*

These results (see proofs in Appendix) guarantee that the number of extrema does not decrease when the scale tends to zero. This aspect constitutes the morphological scale-space monotonicity property.

Theorem 1 (Monotonicity Theorem). *Monotonicity property for the defined scale-space. Let $f : \mathcal{D}_f \subseteq \mathbb{R}^n \to \mathbb{R}$ be a limited function, $g_\sigma : \mathcal{G}_\sigma \subseteq \mathbb{R}^n \to \mathbb{R}$ be a structuring function that satisfies the properties of Proposition 1, and the following set of points $E_{max}(f) = \{\boldsymbol{x} : f(\boldsymbol{x})$ is a local* *maximum}* *and* $E_{min}(f) = \{\boldsymbol{x} : f(\boldsymbol{x})$ *is a local minimum}* *represent the extrema points of* f*. Then, for any* $0 < \sigma_1 < \sigma_2$*,*

$$E_{min}(f \oslash g_{\sigma_2}) \subseteq E_{min}(f \oslash g_{\sigma_1}) \subseteq E_{min}(f), \text{ and}$$
$$E_{max}(f \oslash g_{\sigma_2}) \subseteq E_{max}(f \oslash g_{\sigma_1}) \subseteq E_{max}(f).$$

That is, the number of local maxima (minima) decreases monotonically with the increase of scale [4].

Finally, proposition 3 states that the operator approaches $f(x)$ as the scale parameter approaches zero. In other words, as $\sigma \to 0$, the value of the operator converges to the original image value (see proof in Appendix).

Proposition 3 (Convergence to the original image value)**.** *If the signal* $f(\boldsymbol{x}) : \mathcal{D}_f \subseteq \mathbb{R}^n \to \mathbb{R}$ *is continuous at some* $\boldsymbol{x} \in \mathcal{D}_f$*, then* $(f \oslash g_\sigma)(\boldsymbol{x}) \to f(\boldsymbol{x})$ *as* $\sigma \to 0$*.*

4. Results

First, we give some examples of the binary transformation represented by Equation 4, where segmentation and filtering results are easily obtained. After, we apply the h-maxima transform in the image processed by Equation 3, to further extract markers to be used in a watershed transform.

4.1 Binary results

The first example shows the segmentation of a historical document in which the front side of the paper contains ink components from its verso side. The results were compared against the moving averages [19] algorithm, specially designed for segmenting text images. This method considers a threshold based on the mean gray level of the last n pixels. Figure 3 illustrates the better performance of our operator in the sense that it suppresses properly the components belonging to the reverse side of the paper.

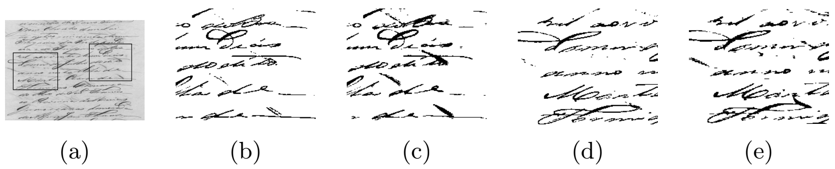

(a) (b) (c) (d) (e)

Figure 3. Segmentation example for a historical document image. (a) Original image, and segmentation results for two select regions using the moving averages algorithm ((c) and (e)) and the proposed operator ((b) and (d)).

The second experiment was carried out based on a set of images with varying lighting conditions (linear, Gaussian and sine-wave) [13], and on a set of well-known threshold-based segmentation methods described in literature, namely, moving averages [19], regional thresholds [13], and the Otsu's [12] thresholding algorithm. An evaluation of these methods as well as the set of considered images can be found in [13]. Figure 4 shows the segmentation results for the mentioned algorithms and for the operator defined by Equation 4.

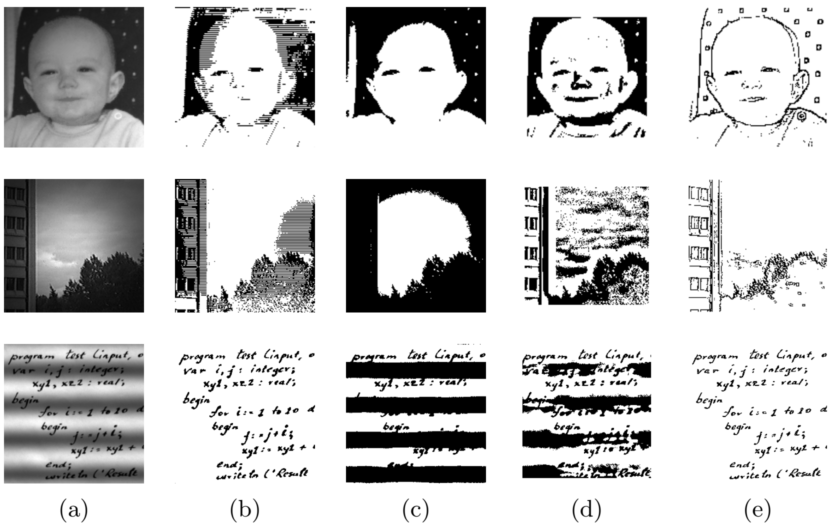


Figure 4. Segmentation results for images with different illumination conditions (linear, Gaussian and sine-wave, respectively). (a) original image, (b) moving averages, (c) Otsu's method, (d) regional thresholding and (e) proposed operator ((i) facel (for $k = 1$ and $\sigma = 0.1$), (ii) skyg (for $k = 1$ and $\sigma = 0.06$) and (iii) pascals (for $k = 2$ and $\sigma = 0.1$).

This well-controlled behavior reflects the results of the previous propositions. The image extrema merge in an organized way, and no new maxima or minima are created, according to the morphological scale-space theory which constitutes the basis of our approach. Finally, note that the operator selects one threshold per pixel based on local features, as it is the case for dynamic thresholdings.

4.2 Gray-scale results

Image segmentation consists basically on partitioning an image into a set of disjoint (non-overlapping) and homogeneous regions which are supposed to correspond to image objects that are meaningful to a certain application. In a morphological framework, this is typically done by first extracting markers of the significant structures, and then using the watershed transform [1] to extract the contours of these structures as accurately as possible.

Although image extrema are frequently used as markers, they can also correspond to insignificant structures or noise. Thus, to prevent the over-segmentation problem, it is necessary to select image extrema according to some criteria, such as contrast, regions area and so forth. A typical approach consists on use the h-maxima (h-minima) transform to suppress all image maxima (minima) whose contrast is lower than a specified value h, and use the extended (regional) extrema as markers.

This notion is closely related to the concept of dynamics, that assigns to each image extrema a value that characterizes the persistence of the structure it marks when applying increasing contrast filters (in other words, the minimal size of the contrast filter for which the extrema is eliminated). Indeed, a regional extrema of an image f is also an extrema of its h-maxima transform only if the structure it marks has contrast higher than h, which also implies a dynamics higher than h [17, 18].

In this paper, we apply the h-maxima transform to select the image maxima that should be used as markers in a watershed transform. The methodology is summarized in the Algorithm 1 below.

Algorithm 1 Segmentation procedure using h-extrema as markers.

```
1: given a height $h$ and an input image $I$;
2: generate a transformed image, $I1$, applying Eq. 3 on $I$;
3: compute the *h-maxima* of $I1$;
4: extract the extended (regional) maxima;
5: compute the watershed transform using these maxima as markers.
```

Quantifying the results of a segmentation algorithm is a challenging task, since this remains a ill-defined problem. Here, we perform a qualitative analysis of the results based on some specific criteria, such as robustness to badly illuminated images. We also compute steps $3-5$ of Algorithm 1 in the original image I, and compare the results against ours.

Figure 5 and Figure 6 show the segmentation results obtained for ill-illuminated images used previously (the parameters are in the Figures' captions). In the last column of each figure, we show the best results obtained using markers from the original image. The markers extracted from the transformed image are less sensitive to illumination problems, yielding a segmentation that enhances the most important structures of the image without introducing artifacts. In Figure 6, we can see that our approach separates correctly the main regions of the figure.

Figure 7 shows the segmentation results, by considering different values of h, for the *road* image transformed by the operator defined in Equation 3 using scale 60 and one iteration. Note that, as the value of h increases, less structures are marked and, thus, less regions are segmented.

Figure 8 shows the segmentation results for the well-known *cameraman* image. As in the previous cases, the quantity of segmented regions depends

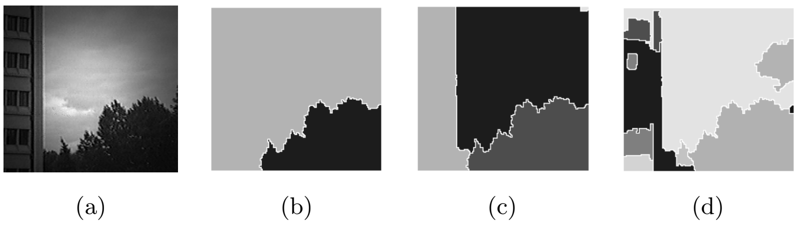


Figure 5. Segmentation results for the *skyg* image. (a) original image, and using transformed images (b) $\sigma = 60$ and $k = 1$, with $h = 2$; (c) $\sigma = 60$ and $k = 1$, with $h = 10$, and (d) based on the original image using $h = 10$.

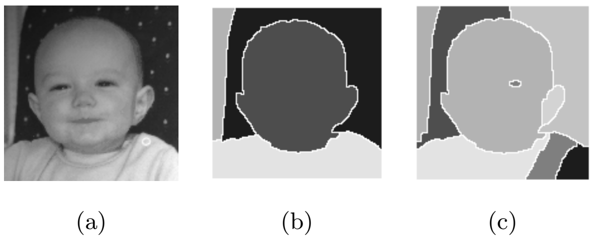


Figure 6. Segmentation results for the *facel* image. (a) original image, (b) using transformed image $\sigma = 10$ and $k = 5$, with $h = 45$, and (c) based on the original image using $h = 45$.

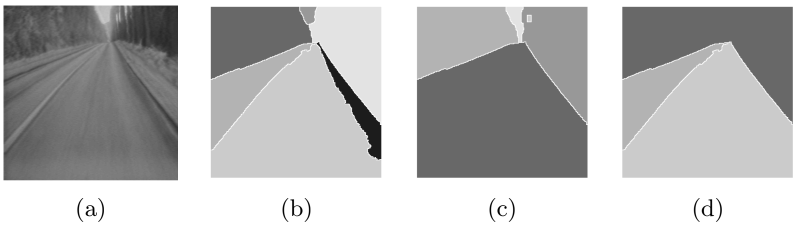


Figure 7. Segmentation results for the *road* image, (a) original image, and using transformed images $\sigma = 60$ and $k = 1$ with (b) $h = 5$, (c) $h = 20$ and (d) $h = 40$.

on the relationship between the σ and h values.

The dynamics' principle can be applied to other families of increasing morphological filters by reconstruction, not only to contrast filters [17]. Figure 9 shows the results obtained for the *lenna* image when including an opening by reconstruction between the steps 3 and 4 of the Algorithm 1. We also make an opening by reconstruction in the original image and further extract the extended maxima to compare the results. Figure 9(b)-(c) show the results for the original image and Figure 9(d)-(e) for the transformed image. In the following, we present some overall conclusions.

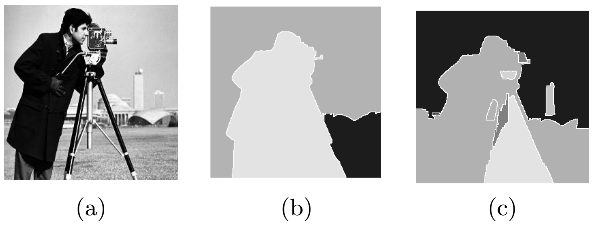


Figure 8. Segmentation results for the *cameraman* image. (a) Original image and using transformed images (b) $\sigma = 15$ and $k = 10$, with $h = 80$ and (c) $\sigma = 31$ and $k = 2$, with $h = 31$.

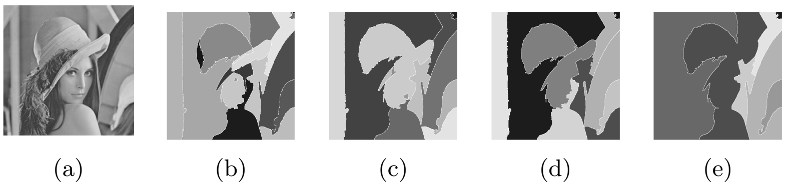


Figure 9. Segmentation results for the *lenna* image. (a) original image, using original image with (b) $h = 11$, (c) $h = 21$, and using transformed images (d) $\sigma = 21$ and $k = 2$, with $h = 21$, and (e) $\sigma = 31$ and $k = 2$, with $h = 31$.

5. Conclusions

In this paper, we explored a new scale-space toggle operator, taking into account the strong monotonicity property for regions of a 2D signal, according to the morphological scale-space theory discussed in literature [4, 20]. The defined operator uses concave structuring functions, which was proved to have contrast properties [14].

In our approach, we deal with transformations of the image maxima and minima at the same time. This aspect, together with the monotonicity property, guarantees that we have an extrema merging simplification that considers the relation between the image extrema along the whole transformation.

We work with an explicit notion of scale guided by the scale-space theory, using a toggle transformation for segmentation problems, unlike the other applications of this operator which consider mainly problems related to image contrast enhancement.

Results comprove the robustness of our approach when dealing with ill-illuminated images. Also, the image simplification obtained using Equation 3 allows a more effective marker extraction.

As a future work, we will deal with problems of locally controlling the extrema merging by taking into account the height of the structuring functions and the distance between the image extrema in the definition of a parametric mapping using both these information.

References

[1] S. Beucher and F. Meyer, *The morphological approach to segmentation: the watershed transformation*, Mathematical Morphology in Image Processing (1993), 433–481.

[2] H. J. A. M. Heijmans and R. van den Boomgaard, *Algebraic Framework for Linear and Morphological Scale-Spaces*, Journal of Visual Communication and Image Representation **13** (2002), 269–301.

[3] P. T. Jackway, *Gradient watershed in morphological scale-space*, IEEE Transactions on Image Processing **15** (1996), 913–921.

[4] P. T. Jackway and M. Deriche, *Scale-space properties of the multiscale morphological dilation-erosion*, IEEE Transactions on Pattern Analysis and Machine Intelligence **18** (1996), 38–51.

[5] H. P. Kramer and J. B. Bruckner, *Iterations of a non-linear transformation for enhancement of digital images*, Pattern Recognition **7** (1975), 53–58.

[6] N. J. Leite and L. E. B. Dorini, *A scaled morphological toggle operator for image transformations*, XIX Brazilian Symposium on Computer Graphics and Image Processing, Proceedings..., 2006, pp. 323–330.

[7] N. J. Leite and M. D. Teixeira, *Morphological scale-space theory for segmentation problems*, IEEE-EURASIP Workshop on Nonlinear Signal and Image Processing, 1999, pp. 364–368.

[8] ______, *An idempotent scale-space approach for morphological segmentation*, Mathematical Morphology and its Applications to Image and Signal Processing, 2000, pp. 291–300.

[9] L. M. Lifshitz and S. M. Pizer, *A multiresolution hierarchical approach to image segmentation based on intensity extrema*, IEEE Transactions on Pattern Analysis and Machine Intelligence **12** (1990), no. 4, 529–540.

[10] J. E. Marsden and M. J. Hoffman, *Elementary Classical Analysis*, Freeman, 1993.

[11] G. Matheron, *Random Sets and Integral Geometry*, John Wiley and Sons, 1975.

[12] N. Otsu, *A threshold selection method from grey-level histograms*, IEEE Transactions on Systems, Man and Cybernetics **9** (1979), no. 1, 377–393.

[13] J. R. Parker, *Algorithms for Image Processing and Computer Vision*, Wiley, 1996.

[14] J. G. M. Schavemaker, M. J. T. Reinders, J. J. Gerbrands, and E. Backer, *Image sharpening by morphological filtering*, Pattern Recognition **33** (2000), 997–1012.

[15] J. Serra and L. Vicent, *An overview of morphological filtering*, Circuits, Systems and Signal Processing **11** (1992), no. 1, 47–108.

[16] P. Soille, *Morphological Image Analysis: Principles and Applications*, Springer-Verlag, 2003.

[17] C. Vachier and L. Vicent, *Valuation of image extrema using alternating filters by reconstruction*, SPIE Neural, Morphological, and Stochastic Methods in Image and Signal Processing, 1995, pp. 94–103.

[18] C. Vachier and F. Meyer, *Extinction value: a new measurement of persistence*, IEEE Workshop on nonlinear signal and image processing, 1995, pp. 254–257.

[19] P. Wellner, *Adaptive thresholding for the digital desk*, Xerox, EPC1993-110, 1993.

[20] A. P. Witkin, *Scale-space filtering: a new approach to multi-scale description*, IEEE International Conference on Acoustics, Speech, and Signal Processing, 1984, pp. 150–153.

Appendix: Proof of the propositions

Proposition 2. First, we prove that the operator defined in Equation 3 performs a dilation on image maximum, that is, $(f \oslash g_\sigma)(\mathbf{x}_{max}) = (f \oplus g_\sigma)(\mathbf{x}_{max})$. To avoid level-shifting and horizontal translation effects, we require that $\sup_{\mathbf{t}\in\mathcal{G}}\{g(\mathbf{t})\} = 0$, and $g(\mathbf{0}) = 0$. Assume that the sup occurs for $t = \zeta$.

$$\begin{aligned}(f \oplus g_\sigma)(\mathbf{x}_{max}) &= \sup_{\mathbf{t}\in\mathcal{N}}\{f(\mathbf{x}_{max} - \mathbf{t}) + g_\sigma(\mathbf{t})\}\\ &= f(\mathbf{x}_{max} - \zeta) + g_\sigma(\zeta)\\ &\le f(\mathbf{x}_{max} - \zeta) \le f(\mathbf{x}_{max}).\end{aligned}$$

Since at $\mathbf{t} = \mathbf{0}$ we have $f(\mathbf{x}_{max}) + g_\sigma(\mathbf{0}) = f(\mathbf{x}_{max})$, it follows that $\sup_{\mathbf{t}\in\mathcal{N}}\{f(\mathbf{x}_{max} - \mathbf{t}) + g_\sigma(\mathbf{t})\} = f(\mathbf{x}_{max})$. The proof for $(f \oslash g_\sigma)(\mathbf{x}_{min}) = (f \ominus g_\sigma)(\mathbf{x}_{min})$ is analogous. Based on this, the final proof of Proposition 2 can be found in [4], as well as the proof of Theorem 1.

Proposition 3. The operator defined in Equation 3 can be easily rewritten as

$$(f \oslash g_\sigma)(\mathbf{x}) = \begin{cases} \psi_1^k(\mathbf{x}) & \text{if } f(\mathbf{x}) > \frac{1}{2}(\beta_n + \alpha_n),\\ f(\mathbf{x}) & \text{if } f(\mathbf{x}) = \frac{1}{2}(\beta_n + \alpha_n),\\ \psi_2^k(\mathbf{x}) & \text{otherwise,}\end{cases}$$

with $\alpha_n = \max\limits_{\mathbf{t}\in N(\mathbf{x},k),\mathbf{t}\neq 0}\{f(\mathbf{x}), f(\mathbf{x}-\mathbf{t}) + (-|\sigma_n|^{-1}))\}$ and $\beta_n = \min\limits_{\mathbf{t}\in N(\mathbf{x},k),\mathbf{t}\neq 0}\{f(\mathbf{x}), f(\mathbf{x}-\mathbf{t}) - (-|\sigma_n|^{-1})\}$, where σ_n is the n-th scale and $N(\mathbf{x}, \epsilon)$ represents the set of pixels located in a chess distance less or equal ϵ from $\mathbf{x}$. The sequence (α_n) satisfies:

- $\alpha_n \ge f(\mathbf{x})$;
- (α_n) is monotonically decreasing, i.e., $\alpha_n \ge \alpha_{n+1} \forall n$;
- (α_n) is stationary, i.e., $\exists n_0 / \forall n \ge n_0, \alpha_n = f(\mathbf{x}_0)$.

The sequence (β_k) satisfies:

- $\beta_n \le f(\mathbf{x})$;
- (β_n) is monotonically increasing, i.e., $\beta_n \le \beta_{n+1} \forall n$;
- (β_n) is stationary, i.e., $\exists n_0 / \forall n \ge n_0, \beta_n = f(\mathbf{x}_0)$.

Let $\gamma_n = \frac{1}{2}(\alpha_n + \beta_n)$. Since the sequences (α_n) and (β_n) are stationary and monotone, we have that the sequence (γ_n) is both monotonic and stationary [10]. This yields a sequence γ_k that is monotonically increasing or decreasing and converges to the original image value after a certain number of iterations.

Comparing morphological levelings constrained by different markers

Konstantinos Karantzalos[1, 2], Demetre Argialas[1] and Nikos Paragios[2]

[1] *Remote Sensing Lab, National Technical University of Athens, Greece*
{karank,argialas}@central.ntua.gr

[2] *Laboratoire de Mathématiques Appliquées aux Systèmes (MAS), Ecole Centrale de Paris, Chatenay-Malabry, France*
nikos.paragios@ecp.fr

Abstract Morphological levelings are powerful operators and possess a number of desired properties for the construction of nonlinear scale space image representations. In this paper, a comparison between levelings constrained by different multiscale markers — namely, reconstruction openings, alternate sequential, isotropic and anisotropic diffusion filters — was performed. For such a comparison a relation between the scales of each marker was established. The evaluation of the simplified images was performed by both qualitative and quantitative measures. Results indicate the characteristics of each scale space representation.

Keywords: filtering, simplification, scale space representations, leveling.

1. Introduction

Since objects in images belong, in the general case, not in a fixed but in many scales, the use of scale space image representations is of fundamental importance for a number of image analysis and computer vision tasks. The concept of Gaussian scale space goes back to the sixties and was first introduced by Iijima [18, 19]. In the western literature and following the ideas of Witkin [20], Koenderink [3] and Lindeberg [4] many possible ways to derive the Gaussian scale space were introduced and respectively many linear multi-scale operators were developed, all of which, though, present the same important drawback: image edges are blurred and new non-semantic objects may appear at coarse scales [11, 12, 20].

Nonlinear operators and nonlinear scale spaces have been studied and following the pioneering work of Perona and Malik [13] there has been a flurry of activity in partial differential equation and anisotropic diffusion filtering techniques [17]. Such approaches either based on diffusions during which the average luminance value is preserved, either based on geometry-driven diffusions, reduce the problems of isotropic filtering but do not eliminate them completely: spurious extrema may still appear [10].

Advanced scale space morphological representations, the levelings, which have been introduced by Meyer [8] and further studied by Matheron [7] and Serra [14], overcome this drawback and possess a number of desired properties for the construction of such representations. Levelings, which are a general class of self-dual morphological filters, are powerful, do not displace contours through scales and are highly dominated by the structure of the markers used for their construction [5, 6, 8–10].

In this paper, a comparison of different operators — namely, the reconstruction openings, the alternate sequential, isotropic and anisotropic diffusion filters — was performed aiming to study the resulting simplified images and describe the characteristics of the each scale space representation. The comparison was based on both qualitative and quantitative evaluation. The later was focused on measurements about the extent of intensity variation and the structural similarity between the reference image and the leveling.

2. Morphological levelings

Following the definitions from [9], one can consider as f_x and f_y the values of a function f at the two pixels x and y and then define the relations: $f_y < f_x$ (f_y is lower than f_x), $f_y \geq f_x$ (f_y is greater or equal than f_x) and $f_y \equiv f_x$ (the similarity between f_x and f_y, which are at level). Based on these relations the zones in an image without inside contours (isophotes, i.e., contour lines with constant brightness values) will be called smooth flat zones.

Two pixels x, y are smoothly linked ($f_x \diamond f_y$) and may belong to the same R-flat zone of a function f if and only if there exists a series of pixels $\{x_0 = x, x_1, x_2, ..., x_n = y\}$ such that they satisfy a symmetrical relation $f_{x_i} \equiv f_{x_{i+1}}$. For equality $f_{x_i} = f_{x_{i+1}}$ the quasi-flat zones are flat. For the symmetrical relation between two neighboring pixels p and q, $f_p \approx f_q$ with $|f_p - f_q| \leq \lambda$, the quasi-flat zones are defined within a maximum difference (slope). Thus, a set X is a smooth zone of an image f if and only if the two values f_x and f_y are smoothly linked ($f_x \diamond f_y$) for any two pixels x and y in X .

Being able to compare the values of 'neighbouring pixels', the more general and powerful class of morphological filters the levelling can be defined. In general they are a particular class of images with fewer contours than a given image f. A function g is a leveling of a function f if and only if

$$f \wedge \delta g \leq g \leq f \vee \varepsilon g,$$

where δ is an extensive operator ($\delta g \geq g$) and ε an anti-extensive one ($\varepsilon g \leq g$).

For the construction of levelings a class $Inter(g, f)$ of functions h is defined, which separates function g and the reference function f. For the function h we have that $h \in Inter(g, f)$ and so $g \wedge f \leq h \leq g \vee f$. Algorithmically and with the use of h, one can 'interpreter' above equation and construct levelings with the following pseudo-code: in cases where $\{h < f\}$, replace the values of h with $f \wedge \delta h$ and in cases where $\{h > f\}$, replace the values of h with $f \vee \varepsilon h$. Equally and in a unique parallel step we have that

$$h = (f \wedge \delta h) \vee \varepsilon h.$$

The algorithm is repeated until the above equation has been satisfied everywhere. This convergence is certain, since the replacements on the values of h are pointwise monotonic. Hence, levelings can be considered as transformations $\Lambda(f, h)$ where a marker h is transformed to a function g, which is a leveling of the reference signal f. Where $\{h < f\}$, h is increased as little as possible until a flat zone is created or function g reach the reference function f and where $\{h > f\}$, h is decreased as little as possible until a flat zone is created or function g reach the reference function f. This makes function g be flat on $\{g < f\}$ and $\{g > f\}$ and the procedure continues until convergence.

2.1 Scale space representations with morphological levelings

Levelings, which are a general class of morphological operators, are powerful and characterized by a number of desirable properties for the construction of nonlinear scale space representations. They satisfy the following properties [8–10]:

(i) the invariance by spatial translation,
(ii) the isotropy, invariance by rotation,
(iii) the invariance to a change of illumination,
(iv) the causality principle,
(v) the maximum principle, excluding the extreme case where g is completely flat.
In addition levelings:
(vi) do not produce new extrema at larger scales,
(vii) enlarge smooth zones,
(viii) they, also, create new smooth zones,
(ix) are particularly robust (strong morphological filters),
(x) do not displace edges.

Above properties make them a very useful simplification tool for a number of low level computer vision applications.

Different types of levelings can be constructed based on different types of extensive δ and anti-extensive ε operators. Based on a family of extensive dilations δ_i and the corresponding family of adjunct erosions ε_i, where $\delta_i < \delta_j$ and $\varepsilon_i > \varepsilon_j$ for $i > j$, multiscale levelings (a hierarchy of levelings) can be constructed [10]. Multiscale levelings can be, also, constructed when the reference function f is associated to a series of marker functions $\{h_1, h_2, ..., h_n\}$. The constructed levelings are respectively

$$g_1 = f,\ g_2 = \Lambda(f, h_1),\ g_3 = \Lambda(f, h_2), ...,\ g_{n+1} = \Lambda(f, h_n).$$

Thus, a series of simpler and simpler images, with fewer and fewer smooth zones are produced. Levelings can be associated to an arbitrary or an alternating family of marker functions. Examples with openings, closings, alternate sequential filters and isotropic markers can be found in [6, 9, 10] and two examples with anisotropic in [15] and [2]. Furthermore, for specific tasks one may take advantage of the knowledge of the scene and design accordingly the family of markers.

3. Comparing different markers for morphological levelings

In this paper, levelings constrained by four different families of markers are compared for the construction of nonlinear scale space representation. The marker functions are constructed by

1. a morphological *reconstruction opening* (RO) with a flat disk-shaped structuring element of radius r (scale parameter), which is the distance from the structuring element origin to the perimeter of the disk,

2. a morphological *alternate sequential filter* (ASF) with reconstruction openings and closings with the same structure element and scale parameter r, as above,

3. an *isotropic* Gaussian function (ISO) with scale parameter σ (standard deviation),

4. an *anisotropic* diffusion filtering (ADF) proposed by Alvarez et al. [1]:

$$\vartheta I(x, y)/\vartheta t = w(|G_\sigma * \nabla I|)|\nabla I| div(\nabla I/|\nabla I|),$$

   where $I(x, y)$ is the original image and t the scale parameter (iterations of the partial differential equation). The term $|\nabla I| div(\nabla I/|\nabla I|)$ diffuses the image $I(x, y)$ in the direction orthogonal to its gradient $|\nabla I|$ and does not diffuse it at all, in the direction of $|\nabla I|$. w is an 'edge-stopping' smooth and non-increasing function like: $w(k) = 1/(1 + k^2/K^2)$ with K a constant. In all cases in this paper, $K = 10$.

3.1 Relation between scales

All the above families of markers are controlled by a scale parameter. For the morphological operators, since scale refers to the same parameter, the comparison can be straight forward, but this is not the case for the levelings that are associated with the isotropic and the anisotropic families of markers.

Towards the establishment of such a scale relation, we have performed extensive experiments by applying these four operators to various scales, attempting first to understand the extent of their filtering effect and relate their result. Observing the extension of their smoothing result and looking forward, in general, to an approximate equivalence, the proposed relation values between the parameters of all four operators were chosen and are shown in Table 1. For example, when four different levelings of scale one are constructed, constrained by the four different families of markers, this means that a disk-shaped structure element with radius $r = 1$ will be used for the two morphological operators, a standard deviation of $\sigma = 1$ will be used for the Gaussian function (in a 5×5 kernel) and 100 iterations t will take place for the computation of the anisotropic marker.

Table 1. Establishing a relation between the scales of the different markers. Proposed values for scales: 1 to 7. The scale parameter for the morphological operators is the size (radius) of the disk-shaped structure element, for the isotropic marker the value of the standard deviation and for the anisotropic the number of the iterations of the partial differential equation.

Leveling's Scale	Values for the scale relation of the four different type of markers			
	Structure element's size r for RO and ASF	Isotropic diffusion		Anisotropic diffusion iterations t
		Standard deviation σ	Kernel size	
1	1	0.5	5×5	100
2	2	1	7×7	200
3	3	1.5	11×11	300
4	4	2	13×13	400
5	5	2.5	17×17	500
6	6	3	19×19	600
7	7	3.5	23×23	700

4. Results and discussion

Levelings (fixed levelings associated to an extensive dilation δ and the adjunct erosion ε) which were associated to RO, ASF, ISO and ADF families of markers, were compared. Two reference images were used for this comparison: an artificial test image and the cameraman test image. The artificial test image was a binary 'chessboard type' one, which was contaminated with both additive and salt and pepper noise. Half was black and half was white, forming a horizontal straight separation/ edge Figure 1.

In addition, for the comparison apart from the qualitative evaluation, a quantitative one also took place, based on three quantitative measures (RMS, NRMS, SSIM and NCD), which are described in Appendix.

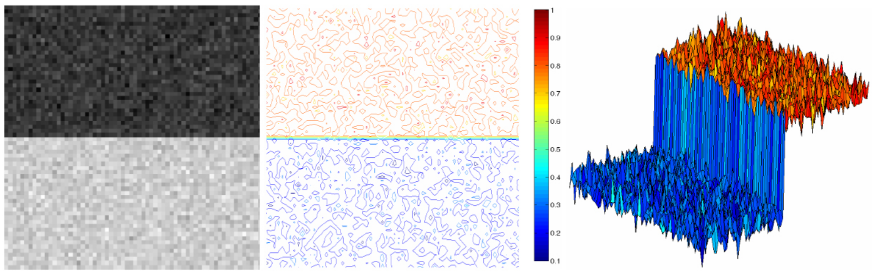


Figure 1. The artificial test image (left), its contours-isophotes, lines of constant brightness (middle) and its 3D representation, in which brightness values are proportional to surface's height.

4.1 Test image

In Figure 2, the resulting levelings (of scale 2) constrained by the four distinct families of markers (RO, ASF, ISO and ADF) are shown. The ASF lead to the most intensive filtering result producing large smooth zones. The RO simplified and at the same time enhanced abrupt brightness changes in a number of small regions. The ADF simplified the image and at the same time preserved regions with strong intensity changes, contrary to ISO. The above qualitative evaluation can be confirmed by the quantitative measures in Table 2. With the ASF, the resulting leveling yielded to the larger RMSE, NMSE values (its brightness values differ much from the original image) and to the smallest SSIM value, that confirms its lower structural similarity with the reference image. The leveling that was constrained by the ADF simplified the image but kept the closest relation with the reference image, regarding both i) the extent of variation in intensity values (RMSE and NMSE measures) and ii) their structural similarity (SSIM measure).

Furthermore, in order to compare, the resulting, from the different levelings simplification (scale 2), cross-sections along the y axis, are shown in Figure 2 (bottom right). First of all one can observe that all methods do not displace edges and in particular the 'black to white region' edge. The constrained by the ASF leveling differed most from the reference image intensities and the leveling constrained by the ISO simplified but at the same time smoothed the brightness values between the different image's zones. Moreover, the leveling constrained by the RO did enhanced the difference in intensities between image zones and those constrained by the ADF simplified and at the same time followed, more constantly, the changes in image intensity values.

4.2 Cameraman image

In Figure 3 (left five images), the resulting levelings (scale 4) constrained by the four (RO, ASF, ISO and ADF) families of markers, are shown which

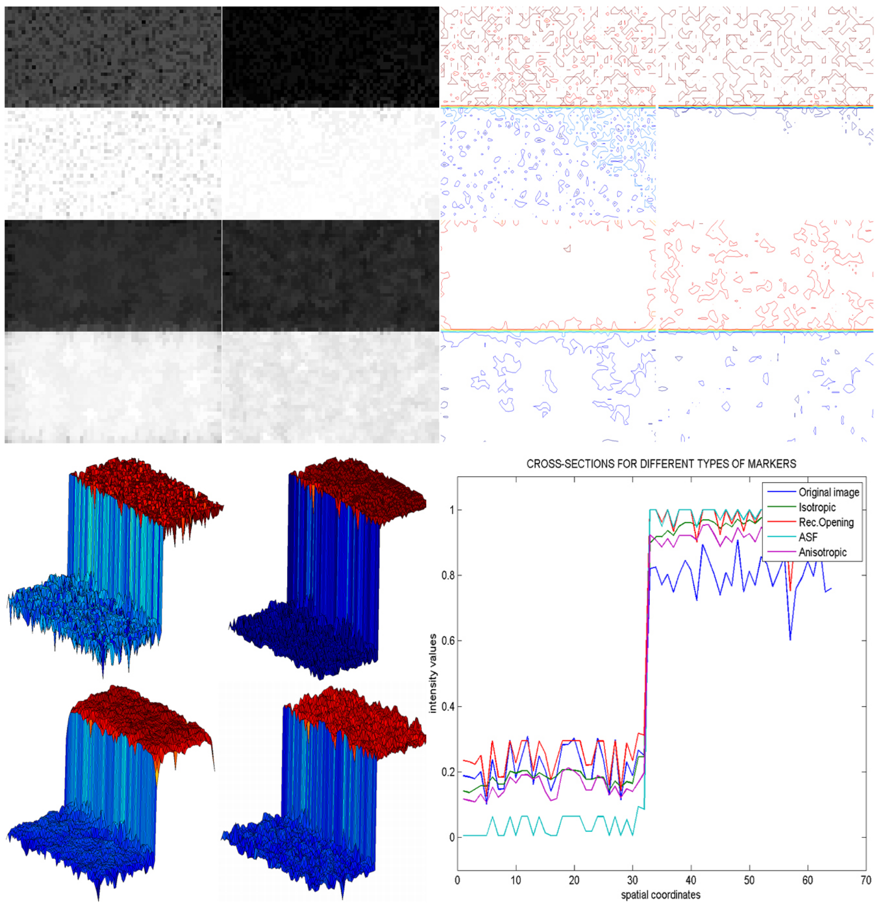


Figure 2. Top Left : Resulting levelings of scale 2 constrained by the four families of markers (RO: top left, ASF: top right, ISO: bottom left and ADF: bottom right). Top Right: The contours of the resulting levelings. Bottom Left: 3D representations, in which brightness values are proportional to surface height. Bottom Right: Line plot with the cross-sections along the y image axis of the compared levelings.

were applied to the cameraman test image. ASF lead to the most intensive filtering result producing large smooth zones. In particular, it suppressed regional extrema in regions with proportional size to the structure element (like the top of the two buildings, in the right center of the image). Similarly, the RO marker was robust in flattening bright regions with proportional size with the structure element (like the bright values in the top of the two buildings in the right center of the image). The ADF marker lead to a simplified image, that is characterized by the preserved level of contrast between the different image flattened zones, contrary to ISO and the other morphological markers. The above qualitative evaluation can be confirmed by the

Table 2. Quantitative evaluation of the resulting levelings by RO, ADF, ISO and ADF markers. Results for the artificial test image (scale 2) and the cameraman test image (scale 4 and scale 7) are presented. In general, the leveling that was constrained by the ADF, did simplified the images and at the same time scored better in all quantitative measures, indicating that it preserves effectively changes in intensities and respects more efficiently the structural similarity with the reference image.

	Quantitative measures	Markers			
		RO	ASF	ISO	ADF
Test image scale 2	RMSE	0.126	0.187	0.118	0.103
	NMSE	0.046	0.101	0.040	0.031
	SSIM	0.9981	0.9956	0.9984	0.9989
Cameraman scale 4	RMSE	13.748	15.649	10.132	4.325
	NMSE	0.011	0.014	0.006	0.001
	SSIM	0.923	0.847	0.904	0.933
Cameraman scale 7	RMSE	20.963	22.652	13.108	4.650
	NMSE	0.024	0.029	0.010	0.001
	SSIM	0.851	0.757	0.866	0.925
Crop of cameraman scale 7	RMSE	30.914	31.943	17.831	2.870
	NMSE	0.053	0.057	0.018	0.001
	SSIM	0.831	0.772	0.891	0.983

quantitative measures in Table 2 (cameraman image, scale 4). Constrained by the RO and ASF, the resulting levelings yielded to the broadest intensity variations, since their brightness values differ much from the original image (RMSE and NMSE measures). In addition, the ASF and the ISO markers produced levelings with the smallest structural similarity with the original image (SSIM). The leveling that was constrained by the ADF, simplified the cameraman image and scored better in all quantitative measures, indicating that it preserves effectively changes in intensities and at the same time in the structural similarity with the reference image.

Furthermore, in Figure 4 (left plot), the cross-sections along a part of the y axis, are shown, from the levelings (scale 4), which were constrained by the RO (red line), the ASF (turquoise line), the ISO (green line) and the ADF (purple line) families of markers. All methods did not displace image edges. The ASF and RO markers resulted into the most extended simplification and drew most away from the reference image intensities. The ADF markers simplified the reference image and at the same time followed more constantly, than all the other markers, intensity changes between the different image zones, due to its edge preserving nature.

In addition, in Figure 3 (five images on the right), the resulting levelings of scale 7, are shown for a crop of the cameraman image. ASF lead to the most intensive filtering result producing large smooth zones and in partic-

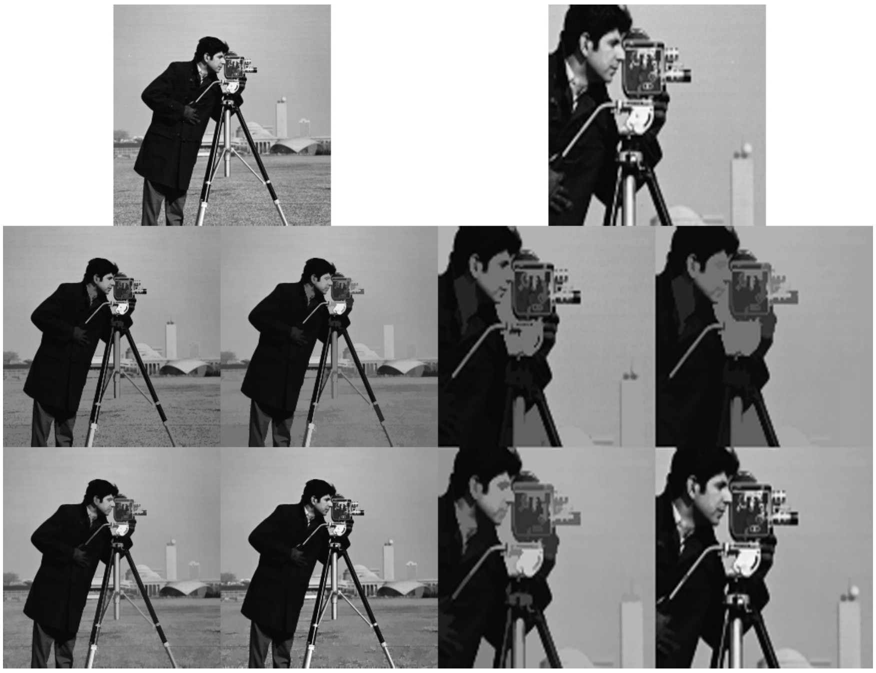


Figure 3. Left five images: The reference cameraman test image (top) and the levelings (scale 4) constrained by the four families of markers (RO: top left, ASF: top right), ISO: bottom left and ADF: bottom right). Right five images: A crop of the cameraman test image and the resulting levelings of scale 7.

ular, eliminated objects with a size proportional to the structure element. Similarly, the RO marker was robust in flattening and in eliminating bright objects with a size proportional to the structure element. The ADF marker lead to a simplified image, that is characterized by the preserved level of contrast between the different image flattened zones, opposite to ISO and the other morphological markers. The resulting leveling from the ADF marker is a simplified version of the reference image on which the edges and the contrast have been preserved.

Quantitative measures in Table 2 (crop of cameraman image, scale 7), indicate that the RO and the ASF levelings resulted into the broadest intensity variations (RMSE and NMSE measures). The ASF, the RO and the ISO markers produced levelings with the smallest structural similarity with the original image (SSIM). The leveling that was constrained by the ADF, scored by far better in all the quantitative measures, indicating that it preserves effectively changes in intensities and at the same time in the structural similarity with the reference image. Finally, in Figure 4 (right), cross-sections along the y axis of the crop of the cameraman image, are shown, from the levelings (scale 7). The ADF marker was the only one which did preserve small changes between the different image zones, due to

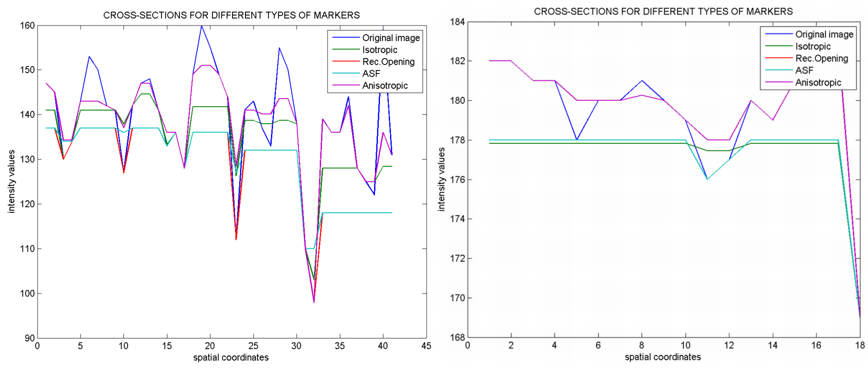


Figure 4. Line plot with the cross-sections of the different levelings for the cameraman image (scale 4, left) and for a crop of the same image (scale 7, right). The simplified brightness values of the different levelings-along a part of the y axis- are shown, which were constrained by RO (red line), ASF (turquoise line), ISO (green line) and ADF (purple line). All methods did not displace image edges. The ASF and RO markers resulted into the most extended simplification and drew most away from the reference image intensities. The ADF markers simplified less and at the same time followed more constantly, than all the other markers, the image intensity changes between the different zones, due to its edge preserving nature.

its edge preserving nature.

5. Conclusions and future perspectives

In this paper, a framework for the comparison of levelings that were constrained by different markers was developed, through the introduction of a relation between the scale parameters of all markers. Four different families of markers were evaluated both by a qualitative and a quantitative comparison of their resulting simplification. The evaluation of the different families of markers concluded to the following points:

- The ASF and RO markers resulted into the most extended simplification and differed most from the reference image intensities.
- The ADF markers yielded to a simplified version of the reference image which followed more constantly, than all the other cases, the intensity changes between the different image zones, due to its edge preserving nature.
- The leveling that was constrained by the ADF, scored by far better (in terms of keeping small the extent of intensity variations and high the structural similarity with the reference image) in all the quantitative measures.
- ASF (and resp. RO) levelings eliminated objects (resp. bright objects) with a size proportional to their structure element. Similarly,

the ISO leveling eliminated — with an isotropic diffusion procedure which, contrary to ADF, blurs image edges — objects according to the standard deviation value.

- The ADF marker lead to a simplified image, which is characterized by the preserved level of contrast between the different image flattened zones, contrary to ISO and the other morphological markers.

Subjects for further research are the establishment of an axiomatic relation between the scales of different markers and their evaluation for specific computer vision tasks like the segmentation and the extraction of specific objects.

Acknowledgments

The project is co-funded by the European Social Fund (75%) and National Resources (25%) — Operational Program for Educational and Vocational Training II (EPEAEK II) and particularly the Program PYTHAGORAS.

References

[1] L. Alvarez, P. L. Lions, and J. M. Morel, *Image selective smoothing and edge detection by nonlinear diffusion. II*, SIAM J. Numer. Anal. **29** (1992), no. 3, 845–866.

[2] K. Karantzalos, D. Argialas, and A. Georgopoulos, *Anisotropic Morphological Levelings*, International Journal of Image and Vision Computing, (under review).

[3] J. J. Koenderink, *The Structure Of Images*, Biological Cybernetics **50** (1984), 363–370.

[4] T. Lindeberg, *Scale-Space Theory in Computer Vision*, Kluwer, 1993.

[5] P. Maragos, *Algebraic and PDE Approaches for Lattice Scale-Spaces with Global Constraints*, International Journal of Computer Vision **52** (2000), 121–137.

[6] P. Maragos and F. Meyer, *Nonlinear PDEs and Numerical Algorithms for Modeling Levelings and Reconstruction Filters*, International Conf. on Scale-Space Theories in Computer Vision (Greece, 1999), Lecture Notes on Computer Science, pp. 363–374.

[7] G. Matheron, *Les Nivellements*, Centre de Morphologie Mathematique, 1997.

[8] F. Meyer, *From connected operators to levelings*, Mathematical Morphology and Its Applications to Image and Signal Processing (H. Heijmans and J. Roerdink, Eds.), Kluwer Academic (1998), 191–198.

[9] ———, *Image Simplification Filters for Segmentation*, International Journal of Mathematical Imaging and Vision **20** (2004), 59–72.

[10] F. Meyer and P. Maragos, *Nonlinear Scale-Space Representation with Morphological Levelings*, J. Visual Communic. and Image Representation **11** (2000), 245–265.

[11] M. Nielsen, P. Johansen, O. F. Olsen, and J. Weickert (Eds.), *Scale-Space Theories in Computer, Lecture Notes in Computer Science*, Springer-Verlag, 1999.

[12] E. J. Pauwels, L. J. Van Gool, P. Fiddelaers, and T. Moons, *An Extended Class of Scale-Invariant and Recursive Scale Space Filters*, IEEE Trans. Pattern Analysis and Machine Intelligence **17** (1995), 691–701.

[13] P. Perona and J. Malik, *Scale-Space and Edge Detection Using Anisotropic Diffusion*, IEEE Trans. Pattern Analysis and Machine Intelligence **12** (1990), 629–639.

[14] J. Serra, *Connections for sets and functions*, Fundamentae Informatica **41** (2000), 147–186.

[15] A. Sofou, G. Evangelopoulos, and P. Maragos, *Soil image segmentation and texture analysis: a computer vision approach*, IEEE Geoscience and Remote Sensing Letters **2** (2005), no. 4, 1–4.

[16] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, *Image Quality Assessment: From Error Visibility to Structural Similarity*, IEEE Transactions on Image Processing **13** (2004), 600–612.

[17] J. Weickert, *Anisotropic Diffusion in Image Processing*, Dept. of Mathematics, University of Kaiserslautern, 1996.

[18] J. Weickert, S. Ishikawa, and A. Imiya, *On the History of Gaussian Scale-Space Axiomatics*, Gaussian scale-space theory (1997), pp. 45–59.

[19] ______, *Linear Scale-Space has First been Proposed in Japan*, J. Math. Imaging Vis. **10** (1999), no. 3, 237–252.

[20] A. P. Witkin, *Scale-Space Filtering*, International Joint Conference on Artificial Intelligence (1983), pp. 1019–1022.

Appendix

Objective methods for assessing perceptual image quality traditionally attempted to quantify the visibility of errors (differences) between a processed image and a reference image using a variety of known properties of the human visual system. In this regard the simplest and most widely used quality metrics are the root mean squared error (RMSE) and the normalized mean square error (NMSE). RMSE is computed by averaging the squared intensity differences of the processed and reference image pixels and NMSE normalized to a range between 0 and 1. Both measures give a quantitative sense for the extent of variation between the intensity values of the two compared images forming a kind of a generalized standard deviation measure. RMSE and NMSE are appealing because they are simple to calculate, have clear physical meanings, and are mathematically convenient in the context of optimization. But they are not very well matched to perceived visual quality [16]. Hence, it has been also adopted a recently proposed alternative complementary quality measure of the structural similarity (SSIM) between two images, which compares local patterns of pixel intensities that have been normalized for luminance and contrast [16]. The above three quality measures (RMSE, NMSE and SSIM) are aiming to an objective image quality assessment of the achieved results.

Leveling cartoons, texture energy markers, and image decomposition

PETROS MARAGOS and GEORGIOS EVANGELOPOULOS

School of Electrical and Computer Engineering, National Technical University of Athens, Greece
{maragos,gevag}@cs.ntua.gr

Abstract The variational $u + v$ model for image decomposition aims at separating the image into a 'cartoon component' u, which consists of relatively flat plateaus for the object regions surrounded by abrupt edges, and a 'texture component' v, which contains smaller-scale oscillations plus possibly noise. Exploiting this model leads to improved performance in several image analysis and computer vision problems. In this paper we propose alternative approaches for $u+v$ decomposition based on levelings and texture energy. First, we propose an efficient method for obtaining a multiscale cartoon component using hierarchies of levelings based on Gaussian scale-space markers. We show that this corresponds to a constrained minimization driven by PDEs and link the leveling cartoons with total variation minimization. Second, we extract the texture component from levelings of the residuals between the image and its multiscale levelings. Further, we employ instantaneous nonlinear operators to estimate the spatial modulation energy in the most active texture frequency bands and use this as a new type of texture markers that yield an improved texture component from the leveling residuals. Finally, we provide experimental results that demonstrate the efficacy of the proposed image decomposition methods.

Keywords: leveling, texture, energy, image decomposition.

1. Introduction

Decomposing an image f into its structural part (objects or geometric features at various scales represented by their regions, boundaries, and mean intensities) and its texture part is both an interesting problem as well as an approach useful for many image analysis and vision applications, such as enhancement, inpainting, segmentation, texture and shape analysis, object description.

A recently proposed method for image decomposition is the $f = u + v$ model, where the u part is called the "cartoon component" and consists of relatively smooth or flat plateaus for the object regions surrounded by abrupt intensity walls, whereas the small-amplitude oscillatory v part is called the "texture". If there is also noise or some other type of insignificant residual w, then a refined model $f = u + v + w$ may be used. Next we summarize some previous works in this area and outline our new contributions.

Background on image decomposition: Many of the nonlinear edge-preserving image smoothing schemes can create cartoon approximations of an image. Examples include the anisotropic diffusion and image selective smoothing. Several of these schemes have been shown in [11] to be special cases or closely related to the Mumford-Shah energy functional [12]

$$E_{MS}(u, C) = \iint_{R \backslash C} (||\nabla u||^2 + \lambda(u - u_0)^2) dxdy + \mu \text{Len(C)}. \tag{1}$$

Actually, one way of obtaining a piecewise-smooth cartoon u from an initial image u_0 is via minimization of the above functional by searching for those u that are *piecewise-constant* on the regions R. This has been used both for denoising and boundary detection. Another approach is via the total variation (TV) image denoising method [14] which finds a cartoon u by minimizing the TV norm $\iint_R ||\nabla u||$ subject to $\iint_R (u - u_0) = 0$ and $\iint_R (u - u_0)^2 = \sigma^2$, or equivalently by minimizing the functional

$$E_{ROF}(u) = \iint_R ||\nabla u|| dxdy + \lambda \iint_R (u - u_0)^2 dxdy, \tag{2}$$

on some image domain R. This minimization is done via a PDE (gradient-descent) solver that finds a local minimum of the TV functional. While the TV approach performs well for edge-preserving image denoising, it may not preserve texture for small λ. Y. Meyer [10] changed the TV optimization problem (2) by using instead of the L_2 norm other norms that are more appropriate to preserve texture. Thus, Meyer introduced the decomposition of an image f into a model $u + v = f$ where u and v result from the modified optimization problem, u is some type of cartoon while v contains the texture (plus possibly noise).

Vese & Osher [16, 17] developed a PDE-based iterative numerical algorithm to estimate the u and v components by approximating Meyer's weaker norms. Texture is assumed to be an oscillating function

$$v = \text{div}(\vec{g}) = \partial_x g_1 + \partial_y g_2, \tag{3}$$

where the vector $\vec{g}$ captures variation in the vertical and horizontal image directions. The component v may exhibit large oscillations, but yields a small metric as measured by the norms $||\vec{g}||_{L^p} = (\iint |\vec{g}|^p)^{1/p}$ For $p \to \infty$, norm L^p approximates L^∞ and thus the norm of Meyer's space of oscillating functions [10]. A three-component $f = u + v + w$ decomposition model was formulated with the (u, v) components derived by minimizing

$$E_{VO}(u, \vec{g}) = \iint\limits_R |\nabla u| dxdy + \lambda \iint\limits_R |f - (u + \text{div}(\vec{g}))|^2 dxdy + \mu ||\vec{g}||_{L_p}, \tag{4}$$

and the residual $f - u - v$ giving image noise w. By letting $\lambda, p \to \infty$, this scheme approximates the initial decomposition proposed by Meyer.

The above ideas and algorithms for $u + v$ image decomposition have been used for improving image restoration [16] and image inpainting by simultaneous filling-in texture and structure in missing image parts [1].

New contributions: Some open research areas in this interesting $u + v$ decomposition space are: (i) Alternative schemes for estimation of a cartoon u and/or texture v component. (ii) Analysis of the information in the u and/or v components. (iii) Exploitation of this decomposition for improving performance in several image analysis and computer vision problems. In this paper we contribute advances in the first two directions inspired by and further utilizing the $u + v$ idea. First, we propose an efficient method for obtaining a cartoon component, possibly at multiple scales, using nonlinear object-oriented smoothing of the leveling type that is driven by PDEs with global scale-space markers obtained from Gaussian diffusion of the image. We also show optimality of this method via a nonlinear constrained minimization. The residuals among consecutive scales of this leveling pyramid provide us with the texture component. Second, we analyze textural information by using instantaneous nonlinear energy-tracking operators that estimate the spatial modulation energy. This energy tracking focuses on the most active texture frequency bands. Third, we propose an alternative new type of markers for the levelings extracting the texture part which are based on texture modulation energy. Finally, we provide experimental results that demonstrate the efficacy of the proposed methods for $u + v$ decomposition.

2. Levelings: Variational problems

Proofs of the following variational formulations can be found in Maragos [7].

Let $u_0(x, y)$ some smooth initial image and $u(x, y, t)$ some scale-space analysis over some compact image domain R with $u(x, y, 0) = u_0(x, y)$. Maximizing the volume functional by keeping invariant the global supremum

$$\max \iint_R u \, dxdy \ \text{ s.t. } \ \bigvee u = \bigvee u_0, \tag{5}$$

has a gradient flow governed by the PDE generating flat dilation by disks:

$$u_t = ||\nabla u||, \quad u(x, y, 0) = u_0(x, y). \tag{6}$$

Similarly, minimizing the volume functional by fixing the global infimum

$$\min \iint_R u \, dxdy \ \text{ s.t. } \ \bigwedge u = \bigwedge u_0, \tag{7}$$

has a gradient flow governed by the isotropic flat erosion PDE:

$$u_t = -||\nabla u||, \quad u(x, y, 0) = u_0(x, y). \tag{8}$$

Imagine now creating a new type of cartoon by starting from a *reference* image $f(x, y)$ consisting of several parts and a *marker* image $M = u_0(x, y)$ (initial seed) intersecting some of these parts and by evolving M toward f in a monotone way such that all evolutions $u(x, y, t)$, $t \geq 0$, satisfy the following partial ordering, $\forall x, y \in R$

$$t_1 < t_2 \Longrightarrow f(x, y) \preceq_f u(x, y, t_2) \preceq_f u(x, y, t_1) \preceq_f u_0(x, y). \tag{9}$$

The partial order $u \preceq_f g$ means that $f \wedge g \leq f \wedge u$ and $f \vee g \geq f \vee u$. Further, if we partition the following regions R^- and R^+ formed by the zero-crossings of $f - u_0$

$$\begin{aligned} R^- &= \{(x, y) : f(x, y) \geq u_0(x, y)\} = \textstyle\bigsqcup_i R_i^-, \\ R^+ &= \{(x, y) : f(x, y) < u_0(x, y)\} = \textstyle\bigsqcup_i R_i^+, \end{aligned} \tag{10}$$

into connected subregions, then the evolution of u is done by maintaining all local maxima and local minima of u_0 inside these subregions R_i^- and R_i^+, respectively:

$$\bigvee_{R_i^-} u = \bigvee_{R_i^-} u_0 \text{ and } \bigwedge_{R_i^+} u = \bigwedge_{R_i^+} u_0, \quad R = (\bigsqcup_i R_i^-) \sqcup (\bigsqcup_i R_i^+), \tag{11}$$

where $\bigsqcup$ denotes disjoint union. Since the order constraint $f \preceq_f u \preceq_f u_0$ implies that $|f - u| \leq |f - u_0|$, the above problem is equivalent to the following constrained minimization

$$\min \iint_R |u - f| dxdy \text{ s.t. } \bigvee_{R_i^-} u = \bigvee_{R_i^-} u_0, \ \bigwedge_{R_i^+} u = \bigwedge_{R_i^+} u_0. \tag{12}$$

Theorem 1. *A gradient flow for the problem (12) is given by the PDE*

$$\begin{aligned} \partial u(x, y, t)/\partial t &= -\text{sign}(u - f)||\nabla u||, \\ u(x, y, 0) &= u_0(x, y). \end{aligned} \tag{13}$$

The PDE (13) was introduced in [9]. It was studied systematically in [6] where it was proved that it has a steady-state $u_\infty(x) = \lim_{t\to\infty} u(x, t)$ which is a *leveling* of f with respect to u_0, denoted by $u_\infty = \Lambda(u_0|f)$.

A leveling g of some image f was defined geometrically in [9] via the property that, the variation of g between any two close neighbor pixels p, q is bracketed by a larger same-sign variation in the reference image f:

$$g(p) > g(q) \Longrightarrow f(p) \geq g(p) > g(q) \geq f(q). \tag{14}$$

In [6] levelings were defined algebraically as fixed points of triphase operators $\lambda(M|f)$ that switch among three phases, an expansion, a contraction, and the reference f. Further, the leveling of f w.r.t. $M = u_0$ can be obtained as the limit of iterations of λ:

$$u_\infty = \Lambda(M|f) \triangleq \lim_{n\to\infty} \lambda^n(M|f) \preceq_f \cdots \lambda(M|f) \preceq_f M = u_0. \tag{15}$$

The simplest choise for λ is $\lambda(M|f) = [f \wedge \delta(M)] \vee \varepsilon(M)$, where δ and ε are disk dilations and erosions.

3. Leveling-based multiscale cartoons

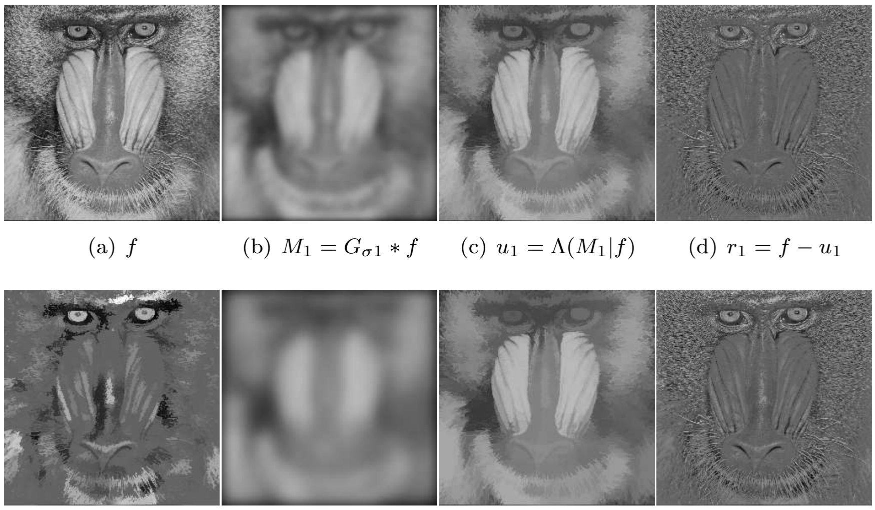
Levelings have many interesting scale-space properties [9]. Due to (9) and (14), they preserve the coupling and sense of variation in neighbor image values, which is good for edge preservation. Further, due to (11) the levelings do not create any new regional maxima or minima. Also, they are increasing and idempotent filters. In practice, they can reconstruct whole image objects with exact preservation of their boundaries and edges. In this reconstruction process they simplify the original image by completely eliminating smaller objects inside which the marker cannot fit. The reference image plays the role of a *global constraint.*

Motivated by all their above attractive properties, we propose an alternative method for u, v decomposition of an original image where we *use the leveling as the cartoon approximation*

$$u = \Lambda(M|f), \tag{16}$$

and its residual $r = f - u$ as containing the texture component v. For u, the marker M plays an important role. Its choice gives us a great flexibility for the final leveling and we could define it based on a multiscale analysis. Specifically, given a reference image f, suppose we can produce various markers M_i, $i = 1, 2, 3, ...$ that are related to some increasing scale parameter i. Then, if we construct the levelings $u_i = \Lambda(M_i|u_{i-1})$, $i = 1, 2, 3, ...$, with $u_0 = f$ the cartoon images u_i constitute a hierarchy of *multiscale levelings* possessing the causality property that u_j is a leveling of u_i for $j > i$. One way to construct such multiscale leveling cartoons is to use a sequence $M_i = f * G_{\sigma i}$ of multiscale markers obtained from sampling a Gaussian scale-space, where G_σ denotes an isotropic 2D Gaussian function of standard deviation σ. As shown in Figure 1, the image edges and boundaries which have been blurred and shifted by the Gaussian scale-space are better preserved across scales by the multiscale levelings. The corresponding residuals texture components $r_i = f - u_i$ contain a hierarchy of multiscale texture components, whose extraction will be detailed in the following sections.

As an alternative to the linear scale-space marker selection, one can consider the use of anisotropic diffusion [13]. At each sequence step the leveling marker is obtained by a version of the image with blurred regions but



(a) f (b) $M_1 = G_{\sigma 1} * f$ (c) $u_1 = \Lambda(M_1|f)$ (d) $r_1 = f - u_1$

(e) $u_1 - u_2$ (f) $M_2 = G_{\sigma 2} * f$ (g) $u_2 = \Lambda(M_2|u_1)$ (h) $r_2 = f - u_2$

Figure 1. Multiscale leveling cartoons and $u+v$ decomposition. **(a)** Reference f. **(b)** Gaus. Marker 1 ($\sigma_1 = 4$). **(c)** Leveling 1 (u_1). **(d)** Residual 1 ($r_1 + 100$). **(e)** Levelings difference $u_1 - u_2$. **(f)** Gaus. Marker 2 ($\sigma_2 = 8$). **(g)** Leveling 2 (u_2). **(h)** Residual 2 ($r_2 + 100$).

adequately preserved boundaries, caused by the constrained diffusion process. Levelings obtained using anisotropic diffusion markers tend to retain information about edges in smaller scales on the cartoon component.

Proposition 1. *Levelings decrease the TV norm:*
(a) If $u = \Lambda(M|f)$, then $\iint ||\nabla u|| \leq \iint ||\nabla f||$.
(b) If $u_i = \Lambda(M_i|u_{i-1})$ with $u_0 = f$, then for all i

$$\iint ||\nabla u_{i+1}|| \leq \iint ||\nabla u_i|| \leq \iint ||\nabla f||. \tag{17}$$

Proof. (a) The levelings create flat plateaus on which the gradient becomes zero. The remaining slopes are the same as for the function. (b) results from (a). □

We can compare our proposed leveling cartoons with the ones derived as the solutions of the TV minimization problem (2) along several directions: (i) The levelings preserve the regional maxima and minima and do not create new ones, while the TV cartoons preserve the global mean value. (ii) The levelings couple and preserve the sense of variation between neighbor pixels (14) whereas the TV cartoons preserve the global variance. (iii) By Proposition 1, the levelings are related to a TV minimization. Further, the TV norm of the leveling cartoon decreases monotonically if we use a hierarchy of multiscale levelings. (iv) The presence of the marker image M gives a leveling cartoon far greater flexibility and multiscale capabilities than the simple regularization constants which control the scale of the TV cartoon.

4. Texture modulation energy

4.1 AM-FM Texture model and energy

Locally narrowband image textures can be modeled as 2D spatial AM-FM signals

$$f(x,y) = a(x,y)\cos[\varphi(x,y)], \quad \vec{\omega}(x,y) = \nabla\varphi(x,y), \tag{18}$$

that are 2D nonstationary sines with a spatially varying amplitude $a(x,y)$ and a spatially-varying instantaneous frequency vector $\vec{\omega}(x,y)$. In particular, the amplitude is used to model local image contrast and the frequency vector contains rich information about the locally emergent spatial frequencies. Thus, it is reasonable to assume that the amplitude $a(x,y)$ and frequency vector $\vec{\omega}(x,y)$ are locally narrowband signals and hence locally smooth. Such modulation models have been proposed by Bovik et al. [2] and Havlicek et al. [4] and have been applied to a variety of image processing and vision problems.

An important problem in modeling image textures with spatial AM-FM signals is to estimate the 2D amplitude and frequency signals using computational vision algorithms that have low complexity and small estimation error. Such an efficient approach was developed in [5] based on an energy operator $\Psi(f) \triangleq ||\nabla f||^2 - f\nabla^2 f$, which is a multidimensional extension of the 1D Teager energy operator. Applying Ψ to a 2D AM-FM signal $f(x,y) = a(x,y)\cos[\varphi(x,y)]$ modeling a texture component yields

$$\Psi[a\cos(\varphi)] \approx a^2||\vec{\omega}||^2, \tag{19}$$

which equals the product of the instantaneous amplitude and frequency magnitude squared and may be called the *texture modulation energy.* The above approximation error is negligible assuming that the instantaneous amplitude and frequency do not vary too fast in space or too greatly in value compared with the carriers. Further, if we also apply the energy operator on the image derivatives $\partial f/\partial x$ and $\partial f/\partial y$, then it is possible to separate the energy into its amplitude and frequency components via a nonlinear algorithm called Energy Separation Algorithm (ESA) [5].

4.2 Multiband texture energy tracking

In Bovik et al. [2] the AM-FM models are not applied directly to the whole (possibly wideband) image. Instead they are used on its bandpass filtered versions that are outputs from a filterbank consisting of 2D Gabor filters. Bandpass filtering isolates highly active texture modulations and has some useful consequences like an increased noise tolerance and the enforcement of some smoothness on the amplitude and frequency signals. The motivation in using Gabor filters is their inherent property to be smooth, compact and attain the lower limit of joint space-frequency resolution uncertainty and model early filtering stages of human vision. The concept of using multiple frequency bands from a bank of bandpass filters for purposes of texture analysis or segmentation has been used with success in previous works, e.g. [2,4,5].

The oscillating functions, indicated by Meyer [10] to model and extract the v component, can be sought via AM-FM image modeling that reveals modulations and the existence of contrast and spatial frequency oscillations. Motivated by the analogies between the modulation models and Meyer's indications we aim to capture oscillatory textural energy by a modified energy operator, through a multiband filtering-modulation based process.

In our work the extracted textured part is filtered through a bank of 2D Gabor filters , which are characterized by impulse response of the form $h_k\left(x,y\right) = e^{-\alpha^2 x^2 - \beta^2 y^2} \cos\left(\Omega_{k1}x + \Omega_{k2}y\right)$, where $\alpha/2\pi, \beta/2\pi$ are the rms bandwidths in each dimension and $(\Omega_{k1}, \Omega_{k2})$ is the k -th filter's central frequency pair. The filters are uniformly arranged in the spatial frequency domain, in a polar wavelet-like tessellation, with equal and directional symmetric bandwidths and cover densely the frequency domain.

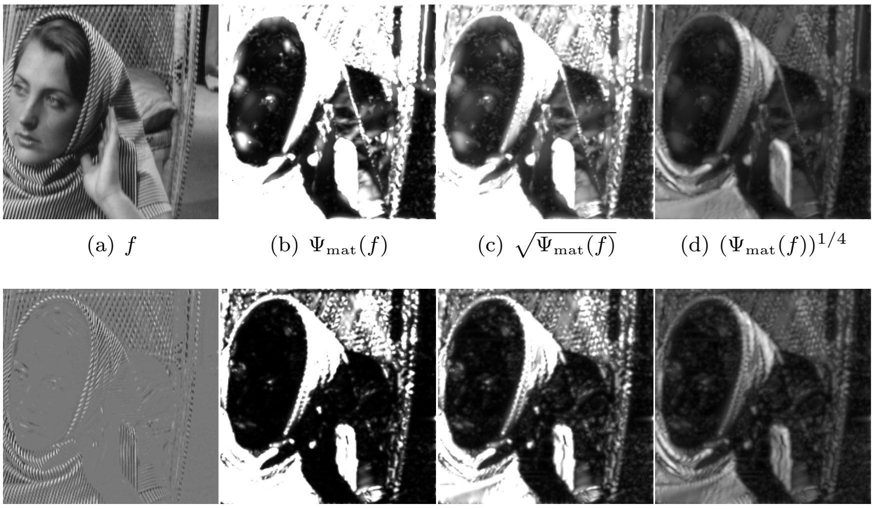
The filtered texture components from each filter output are then averaged by a local averaging filter h_a and the 2D Energy Operator Ψ is applied. We keep the value of the filter with the *Maximum Average Teager (MAT) Energy* per pixel, given by

$$\Psi_{\text{mat}}(v(x,y)) = \arg\max_k \Psi[((v * h_k) * h_a)(x,y)]$$

($*$ denotes convolution), as a means of tracking the most active texture component. The derived Ψ_{mat} is a slowly-varying indication of texture modulation energy, which can classify among different energy levels. It provides both local and global texture information and applied to the level-free v component, $\Psi_{\text{mat}}(v)$ is tracking the most active texture components along multiple modulation bands. Efficient discrete schemes exist for the numerical implementation of the 2D energy operator [3,5].

The above ideas for texture modeling have been used for geometric active contour-based texture segmentation in Evangelopoulos et al [3]. In addition, the $u+v$ image decomposition using levelings for u and texture energy from v was used in a coupled watershed plus texture PDE-based segmentation scheme [15].

In Figure 2 we explore the efficacy of finding dominant components via the MAT Energy for detecting texture areas. This procedure can provide us with *texture energy markers*. We observe that the square root of MAT energy (or even better its higher roots) successfully mark texture areas. Further, notice the lack of image structure and structure features (e.g. edges, blobs) in the markers extracted from the leveling residual r_1.



(a) f (b) $\Psi_{\text{mat}}(f)$ (c) $\sqrt{\Psi_{\text{mat}}(f)}$ (d) $(\Psi_{\text{mat}}(f))^{1/4}$

(e) $r_1 = f - u_1$ (f) $\Psi_{\text{mat}}(r_1)$ (g) $\sqrt{\Psi_{\text{mat}}(r_1)}$ (h) $(\Psi_{\text{mat}}(r_1))^{1/4}$

Figure 2. Texture energy markers and texture detection. Application of the Ψ_{mat} operator and its variants on initial f and the first leveling residual $r_1 = f - u_1$, where $u_1 = \Lambda(f * G_{\sigma 1}|f)$ with $\sigma_1 = 2$. For display, images in (b),(c) and (f),(g) are shown by upper thresholding the energy range at its average value and then linearly stretching onto [0,1].

5. Experiments on image decomposition

Motivated by the ability of the levelings to yield the cartoon component u, either at the first or at the second marker scale, we experimentally investigate in this section two possible schemes to extract the texture component v from the residual r between the image and its leveling.

The first approach uses Gaussian markers both on the original image f to yield the cartoon u (at second scale) as well as on the first residual to yield the texture component according to the algorithm

$$\begin{aligned} u_1 &= \Lambda(M_1|f), \quad M_1 = f * G_{\sigma 1}, \\ u &= u_2 = \Lambda(M_2|u_1), \quad M_2 = f * G_{\sigma 2}, \\ r_1 &= f - u_1, \\ u_r &= \Lambda(M_3|r_1), \quad M_3 = r_1 * G_{\sigma 3}, \quad \sigma_3 = \sigma_1/2, \\ v &= r_1 - u_r. \end{aligned} \tag{20}$$

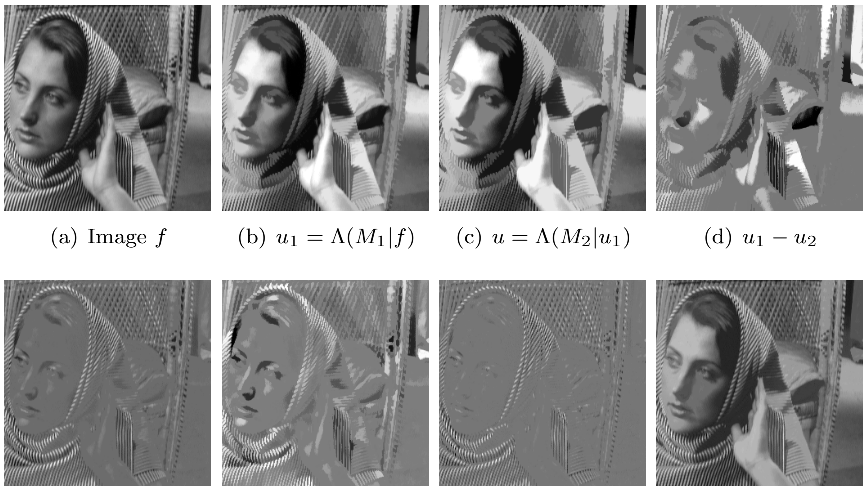

(a) Image f
(b) $u_1 = \Lambda(M_1|f)$
(c) $u = \Lambda(M_2|u_1)$
(d) $u_1 - u_2$
(e) $r_1 = f - u_1$
(f) $u_r = \Lambda(M_3|r_1)$
(g) $v = r_1 - u_r$
(h) $u + v$

Figure 3. Multiscale leveling cartoon and $u + v$ decomposition. Top row: (a) Image f, (b) Leveling 1 (u_1) with Gauss marker $f * G_{\sigma_1}$ ($\sigma_1 = 5$), (c) Cartoon/ Leveling 2 (u_2) with ($\sigma_2 = 10$), (d) Leveling difference ($u_2 - u_1$). Bottom row: (e) Residual $r_1 = f_1 - u_1$, (f) Leveling of r_1 with Gauss marker ($\sigma_3 = \sigma_1/2$), (g) Texture / residual ($v = r_1 - u_r$), (h) Reconstruction ($u + v$).

Figure 3 shows the results of the above algorithm. Figure 4 compares them with the Vese-Osher approach. In this comparison, we note the following.

(i) Decompositions may not be comparable, are not optimum, only made with same L_2 norms on estimated v. (This equality of norms on the two texture components was enforced for purposes of comparison).

(ii) Leveling u_Λ is sharper, yields clearer figure and large scale boundaries, u_{VO} is somehow smoother with smeared less-sharp edges.

(iii) The previous advantage of the leveling cartoon has a tradeoff with the structure kept and evident in w_Λ.

(iv) Smaller-scale structural details (e.g., facial characteristics) are preserved in u_Λ.

(v) Texture components seem similar.

(vi) More texture remains in w_{VO} than on w_Λ, though the latter has kept more structure.

(vii) The reconstruction $u_\Lambda + v_\Lambda$ from levelings tends to 'quantize' the intensity values.

(viii) What kind of residual is this ‘noise’ w_{VO}? Is it really modeling image noise or something else? There is actually a mathematic formula for it, since three levelings are required in total to produce it:

$$w_\Lambda = f - u - v = \Lambda(M_1|f) - \Lambda[M_2|\Lambda(M1|f)] + \Lambda[M_3|f - \Lambda(M_1|f)]. \quad (21)$$

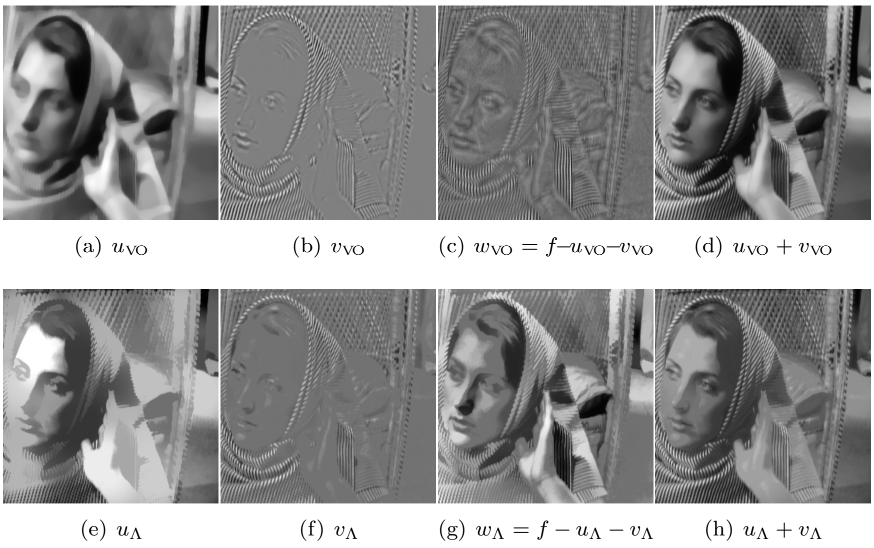
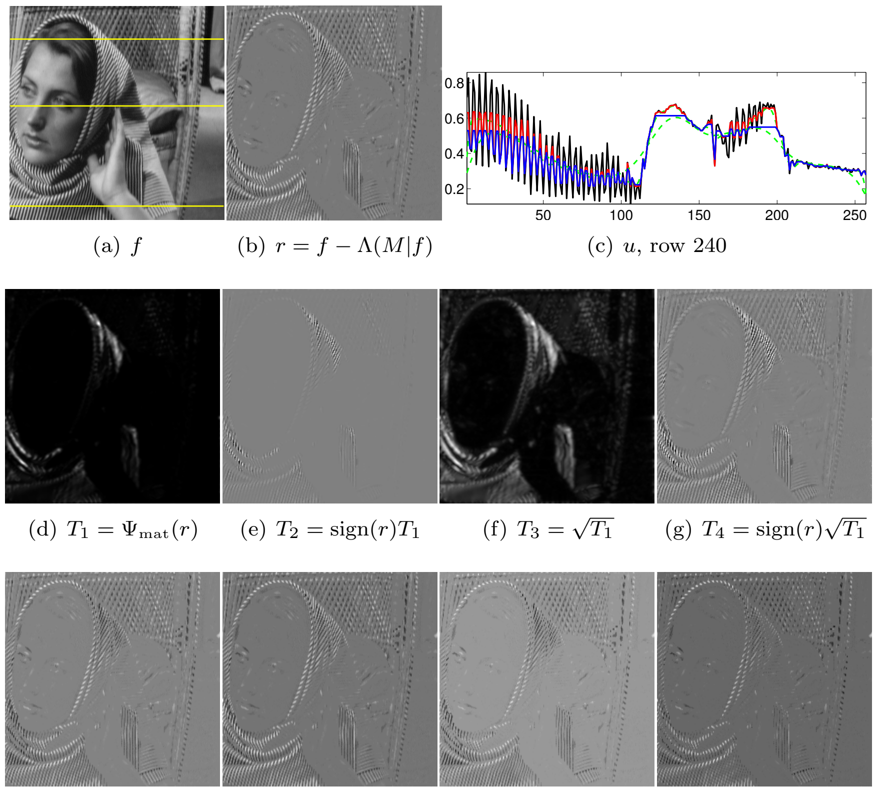

(a) u_{VO}
(b) v_{VO}
(c) $w_{\mathrm{VO}} = f – u_{\mathrm{VO}} – v_{\mathrm{VO}}$
(d) $u_{\mathrm{VO}} + v_{\mathrm{VO}}$
(e) u_Λ
(f) v_Λ
(g) $w_\Lambda = f - u_\Lambda - v_\Lambda$
(h) $u_\Lambda + v_\Lambda$

Figure 4. Comparisons with Vese-Osher $u + v$, left to right cartoon u, texture v, residual noise $w = f - u - v$ and image reconstruction from the model $u + v$. Top row: Vese-Osher algorithm with parameters $(\lambda_{\mathrm{VO}}, \mu_{\mathrm{VO}}) = (5, 0.1)$. Bottom row: Leveling decomposition with Gaussian markers $(\sigma_1, \sigma_2) = (10, 16)$. Parameters for both schemes were chosen so that the texture components have almost equal L_2 norms, i.e $||v_{\mathrm{VO}}||_2 = ||v_\Lambda||_2$. All image values are stretched at full grayscale for display

Next we propose an alternative approach that uses the same algorithm for the cartoon u but derives the texture v by applying a leveling on the cartoon residual based on some type of texture energy markers:

$$v = r_1 - \Lambda(\pm\Psi_{mat}(r_1)|r_1). \quad (22)$$

Figure 5 shows several choices for such energy markers. In our experiments, the best results visually were achieved by using as marker a signed version of the MAT energy of the residual (or possibly its square root). This can be observed both in the resulting texture image component v and its profiles (no slow variation is left on v and the result seems zero-mean).



(a) f (b) $r = f - \Lambda(M|f)$ (c) u, row 240

(d) $T_1 = \Psi_{\text{mat}}(r)$ (e) $T_2 = \text{sign}(r)T_1$ (f) $T_3 = \sqrt{T_1}$ (g) $T_4 = \text{sign}(r)\sqrt{T_1}$

(h) $v = r - \Lambda(T_1|r)$ (i) $v = r - \Lambda(T_2|r)$ (j) $v = r - \Lambda(T_3|r)$ (k) $v = r - \Lambda(T_4|r)$

Figure 5. Markers T based on texture energy and leveling $\Lambda(T|r)$ of cartoon residual $r = f - \Lambda(M|f)$, where $M = f * G_\sigma$. Top row: image f, cartoon residual r and profile of row 240 (black: f, red: r, blue: u, green: marker). Middle row: Texture markers extracted from residual. Bottom row: Final texture components.

6. Conclusions

For the purpose of $u + v$ image decomposition, we have proposed hierarchical levelings based on Gaussian scale-space markers as a candidate model for image cartoons u. This was theoretically and experimentally supported. Further, we provided a viable approach to extract the texture part v based on levelings of the cartoon residuals. An improved version of the texture estimation resulted by performing energy-based dominant component analysis among multiple frequency bands and using this to create as texture detection markers for the levelings of the residuals.

There are numerous applications of the above ideas and algorithms. An ongoing research involves the restoration of ancient wallpaintings from cracks and missing parts by performing the previous $u+v$ decomposition, from which the resulting u part achieves a significant degree of inpainting of the wide cracks and holes in these images, whereas the texture part contains the thin crack lines to be exploited by some further processing. An example of this application is illustrated in Figure 6. There we see that the leveling-based cartoon u is sharper and its corresponding texture v (based on energy markers) contains less structure than previous approaches.

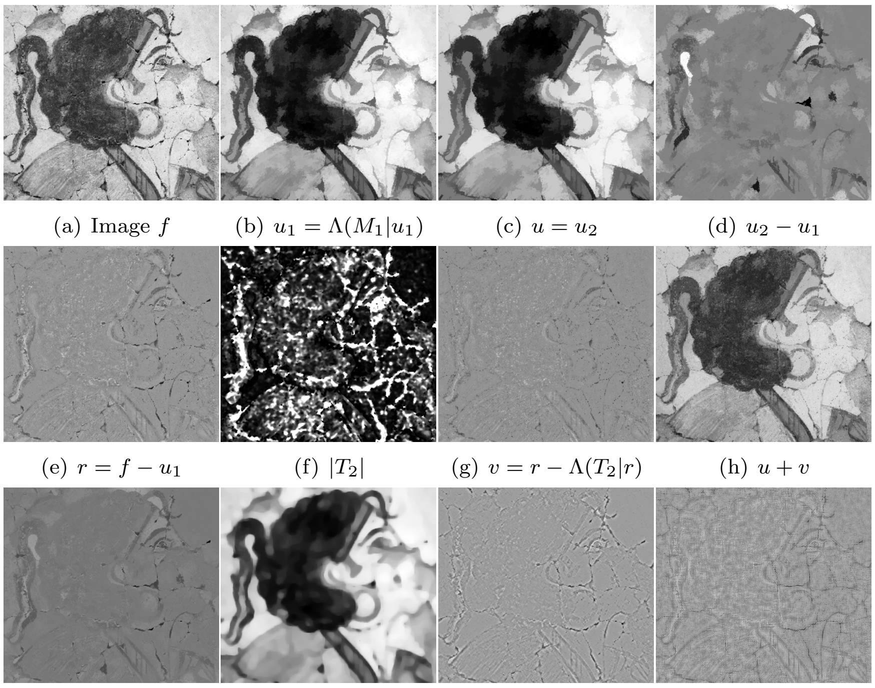


Figure 6. Leveling $u+v$ decomposition for image restoration. Top row: (a) Image f, 300×330 pixels from (6:1) subsampled "Potnia" wallpainting (*prehistoric Thira Acrotiri*), (b) Leveling 1 (u_1) with marker $f{*}G_\sigma$, where G_{σ_1} Gaussian ($\sigma_1 = 4$), (c) Cartoon/ Leveling 2 (u_2), with ($\sigma_2 = 8$), (d) Residual of levelings ($u_2{-}u_1$). Middle row: (e) Residual $r{=}f{-}u_1$, (f) Texture energy ($T = \Psi_{\mathrm{mat}}(r)$) used for marker $T_2 = \mathrm{sign}(r)\sqrt{T}$, (g) Texture ($v = r - \Lambda(T_2|r)$), (h) Reconstruction ($v{+}u$), (i) Modeling error/fidelity, (j),(k),(l) Vese-Osher ($u_{\mathrm{VO}}, v_{\mathrm{VO}}$) and fidelity ($f - u_{\mathrm{VO}} - v_{\mathrm{VO}}$) with ($\lambda_{\mathrm{VO}}, \mu_{\mathrm{VO}}$) = $(5, 0.1)$.

Acknowledgments

This work was supported by the European Network of Excellence MUSCLE and by grants ΠΕΝΕΔ-2003 ΕΔ865 and ΠΕΝΕΔ-2001 [co-financed by E.U.-European Social Fund (75%) and the Greek Ministry of Development-GSRT (25%)]. We thank G. Papandreou and S. Lefkimmiatis for help with pdfLaTeX.

References

[1] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, *Image filling-ing in a decomposition space*, Intl. Conf. on Image Processing, November 2003.

[2] A. C. Bovik, N. Gopal, T. Emmoth, and A. Restrepo, *Localized measurement of emergent image frequencies by Gabor wavelets*, IEEE Trans. Inform. Theory **38** (March 1992), no. 3, 691–712.

[3] G. Evangelopoulos, I. Kokkinos, and P. Maragos, *Advances in Variational Image Segmentation Using AM-FM Models: Regularized Demodulation and Probabilistic Cue Integration*, Proc. VLSM 2005: Springer LNCS 3275, 2005, pp. 121–136.

[4] J. P. Havlicek, D. S. Harding, and A. C. Bovik, *Multidimensional quasi-eigenfunction approximations and multicomponent AM-FM models*, IEEE Trans. Image Proc. **9** (February 2000), no. 2, 227–242.

[5] P. Maragos and A. C. Bovik, *Image Demodulation using Multidimensional Energy Separation*, J. Opt. Soc. Amer. A **12** (September 1995), no. 9, 1867–1876.

[6] P. Maragos, *Algebraic and PDE Approaches for Lattice Scale-Spaces with Global Constraints*, Int'l J. Computer Vision **52** (2003), no. 2-3, 121–137.

[7] ———, *A Variational Formulation of PDEs for Dilations and Levelings*, Proc. Int'l Symp. Math. Morphology (ISMM 2005), 2005.

[8] F. Meyer, *The Levelings*, Proc. 4th Int'l Symp. Math. Morphology and its Applications to image processing, June 1998, pp. 199 –206.

[9] F. Meyer and P. Maragos, *Nonlinear Scale-Space Representation with Morphological Levelings*, J. Visual Communication and Image Representation **11** (June 2000), no. 2, 245–265.

[10] Y. Meyer, *Oscillating Patterns in Image Processing and Nonlinear Evolution Equations*, University Lecture Series, vol. 22, AMS, 2001.

[11] J. M. Morel and S. Solimini, *Variational Methods in Image Segmentation*, Birkhauser, Boston, 1995.

[12] D. Mumford and J. Shah, *Optimal Approximations by Piecewise Smooth Functions and Associated Variational Problems*, Communic. Pure and Applied Math. **42** (1989), no. 5, 577–685.

[13] P. Perona and J. Malik, *Scale-Space and Edge Detection Using Anisotropic Diffusion*, IEEE Trans. Pattern Anal. Mach. Intell. **12** (1990), no. 7, 629–639.

[14] L. Rudin, S. Osher, and E. Fatemi, *Nonlinear Total Variation based Noise Removal Algorithms*, Physica D: Nonlinear Phenomena **60** (1992), no. 1-4, 259–268.

[15] A. Sofou, G. Evangelopoulos, and P. Maragos, *Coupled Geometric and Texture PDE-based Segmentation*, Proc. Int'l Conf. Image Processing, 2005, pp. II–650–3.

[16] L. A. Vese and S. J. Osher, *Modeling textures with total variation minimization and oscillating patterns in image processing*, SIAM J. Scientific Computing **19** (2003), no. 1-3, 553–572.

[17] ———, *Image Denoising and Decomposition with Total Variation Minimization and Oscillatory Functions*, J. Math. Imaging and Vision **20** (2004), 7–18.

IV

IMAGE PROCESSING

A new shape descriptor based on tensor scale

Fernanda A. Andaló, Paulo A. V. Miranda,
Ricardo da S. Torres and Alexandre X. Falcão

Instituto de Computação (IC), Universidade Estadual de Campinas (Unicamp), SP, Brazil
{feandalo,paulo.miranda,rtorres,afalcao}@ic.unicamp.br

Abstract Tensor scale is a morphometric parameter that unifies the representation of local structure thickness, orientation, and anisotropy, which can be used in several computer vision and image processing tasks. In this paper, we exploit this concept for binary images and propose a shape descriptor that encodes region and contour properties in a very efficient way. Experimental results are provided, showing the effectiveness of the proposed descriptor, when compared to other relevant shape descriptors, with regard to their use in content-based image retrieval systems.

Keywords: image processing, shape description, image retrieval.

1. Introduction

The recent growth of the World Wide Web and the new technologies that became available for image acquisition and storage have increased the demand for image retrieval systems based on image properties.

In content-based image retrieval (CBIR) systems, image processing techniques are used to describe the image content, encoding image properties that are relevant to the query. Usually, these properties are represented by shape, color, and texture descriptors of objects or regions within the image. A descriptor can be characterized by two functions: a feature vector extraction function and a similarity function. The feature vector represents the properties extracted from the image and the similarity function computes the similarity between images based on their feature vectors [10].

The shape of an object is an important and basic visual feature for describing image content [14]. Shape representation generally aims at effective and perceptually important shape features based on boundary information – *contour-based methods* – and/or on interior content – *region-based methods*. Each class can be further broken into *structural* and *global approaches*, depending on whether the shape is represented as a whole or by segments or sections [14].

In this work, we propose a new descriptor based on *tensor scale* that exploits region and contour information. Tensor scale [9] is a morphometric parameter yielding a unique representation of local structure thickness, orientation, and anisotropy. That is, at any image point, its tensor scale is represented by the largest ellipse (2D), or ellipsoid (3D), centered at that point and within the same homogeneous region.

We exploit the tensor scale concept for objects and, for sake of simplicity, we only consider objects with a single contour. The descriptor computes the tensor scale ellipse for every object point, divides the object's contour into a predefined number of segments, computes the *influence zone* of each segment and assigns, to each segment, the weighted angular mean orientation [5] of the ellipses within its influence zone. The influence zone of a segment consists of the object pixels that are closest to pixels of that segment than to any other pixel along the contour.

By dividing the contour into a small number of ordered segments, we are aiming at efficiency in encoding contour information. By mapping tensor scale orientation onto the segments, we are exploiting region information. As we will show, this makes the proposed descriptor compact, efficient, and effective for CBIR.

A previous shape descriptor based on tensor scale – Tensor Scale Descriptor [6] (TSD) – was proposed based on the histogram of the tensor scale orientation of the ellipses. Our descriptor introduces a totally new way of exploiting tensor scale orientation, which includes spatial information. We also present a much faster computation of the ellipses by exploiting the Image Foresting Transform (IFT) [2].

2. Background

In [9], Punam introduced a local method for gray-scale images – tensor scale – represented by the largest ellipse within the same homogeneous region, centered at a given pixel p. This method defines the ellipse by three factors:

- $Orientation(p)$ = angle between $t_1(p)$ and the horizontal axis;
- $Anisotropy(p) = \sqrt{1 - \frac{|t_2(p)|^2}{|t_1(p)|^2}}$;
- $Thickness(p) = |t_2(p)|$;

where $|t_1(p)|$ and $|t_2(p)|$, with $|t_1(p)| \geq |t_2(p)|$, denote the length of the two semi-axis of the ellipse centered at p.

A tensor scale ellipse is calculated from sample lines that are traced around a given pixel, from 0 to 179 degrees (Figure 1(a)). The axes of the ellipse are determined by computing the image intensities along each of the sample lines and the location of two opposite edge points on these lines

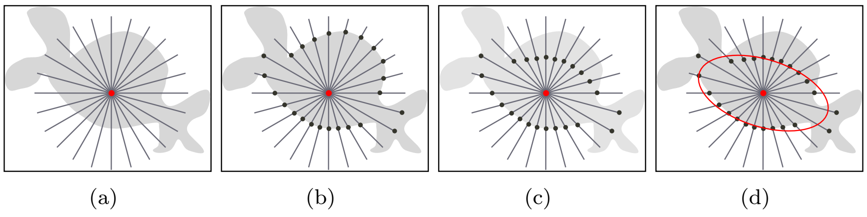


Figure 1. Tensor scale computation.

(Figure 1(b)). The next step consists of repositioning the edge locations to points equidistant to that given pixel, following the axial symmetry of the ellipse (Figure 1(c)). The computation of the best-fit ellipse to the repositioned edge locations is done by Principal Component Analysis (PCA) (Figure 1(d)).

These computations are performed for every pixel of the image. A critical drawback of Punam's approach is that the computational cost of the algorithm makes his method quite prohibitive for CBIR systems. For this reason, Miranda et al. [6] proposed an efficient implementation of the original method, which differs in the following aspects.

The first change was in the edge location phase, in which Miranda's approach is to go along each pair of opposite segments, alternately and at the same time, instead of going along one entire segment by turn. By doing this, the reposition phase is no longer necessary. The second change was the use of two connected thresholds to improve and simplify the original method of detecting edges. The third and final change was the improvement of the ellipse computation phase. Miranda et al. proposed a function g (Equation 1) that gives the angle of the ellipse directly, instead of using PCA. The ellipse orientation is obtained from the value of γ that minimizes the function g below.

$$g(\gamma) = \sum_{i=1,2,\ldots,2m} [x_{i_\gamma}^2 - y_{i_\gamma}^2], \tag{1}$$

where $x_{i_\gamma} = x_i \cos(\gamma) - y_i \sin(\gamma)$, $y_{i_\gamma} = x_i \sin(\gamma) + y_i \cos(\gamma)$, (x_i, y_i) are the relative coordinates of the edge points with respect to the center pixel $p = (x_p, y_p)$ of the ellipse, and $(x_{i_\gamma}, y_{i_\gamma})$ are the new coordinates after applying a rotation by the angle γ.

Considering these optimizations, Miranda et al. [6] proposed the Tensor Scale Descriptor (TSD) for gray-scale images. The idea of their shape descriptor stemmed from the observation that distinct objects often present different tensor scale local orientation distributions of their shape (this is also valid for texture, i.e., their descriptor could be easily extended for colored images). The TSD descriptor computes the tensor scale parameters for the original image and then computes the local orientation histogram, used as feature vector. The matching of two given images by TSD is made by taking the absolute difference of the area between their orientation histograms, after correcting the displacement by correlation (i.e., object rotations cause shifts in the histogram).

In the next sections, we propose a new shape descriptor based on tensor scale – Tensor Scale Descriptor with Influence Zones (TSDIZ) – and show how it can provide considerable improvements in CBIR. We also provide a faster computation of tensor scale, as compared to previous approaches [6,9], by exploiting the Euclidean Image-Foresting Transform [12].

3. The TSDIZ descriptor

The key idea of the proposed descriptor is to map the tensor scale orientations inside an object onto a few segments of its contour, and use this information for shape description.

Orientation mapping is done by exploiting the discrete Voronoi regions (influence zones) of contour segments inside the object. The discrete Voronoi regions can be efficiently obtained by label propagation using the Image Foresting Transform (IFT) [2].

The IFT is a graph-based approach to the design of image processing operators based on connectivity, in which the images are represented by graphs – the pixels are considered as nodes and the arcs are defined by an adjacency relation between pixels. For a given seed set, each seed defines an influence zone consisting of the pixels that are “more closely connected” to that seed than to any other, according to a path-cost function [2]. We use a path-cost function that assigns the closest Euclidean distance between object pixels and contour pixels to each pixel inside the object (Euclidean IFT – Euclidean distance transform via IFT).

The TSDIZ approach divides the contour into segments, labels each contour segment with a distinct number, and propagates these labels inside the object via Euclidean IFT. It is assigned to each segment, a weighted angular mean orientation of the ellipses inside its influence zone, using their anisotropies as weights.

In the next section, we present the Euclidean IFT that is used for tensor scale computation and tensor scale mapping.

3.1 Euclidean IFT

The Euclidean IFT is used for two purposes in TSDIZ: faster tensor scale computation (Section 3.2) and tensor scale orientation mapping (Section 3.3). The advantages of calculating the Euclidean Distance Transform via IFT is that label propagation is executed on-the-fly and in linear time.

In the Euclidean IFT (Algorithm 1), the path-cost function is such that the cost of a path from a seed s to a pixel t in the forest is the Euclidean distance between s and t. The algorithm also needs an Euclidean relation A that is defined as

$$q \in A(p) \Rightarrow (x_q - x_p)^2 + (y_q - y_p)^2 \leq \rho^2, \qquad (2)$$

where ρ is the adjacency radius and (x_i, y_i) are the coordinates of a pixel i in the image.

Our Euclidean IFT assigns three attributes to each object pixel p: the squared Euclidean distance $C(p)$ between p and its closest point s in the contour (forming an optimum cost map), its closest seed $R(p) = s$ (forming a root map), and the label $L(p) = L(s)$ of the segment that contains s (forming a label map).

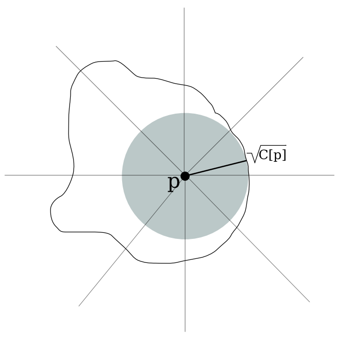
Algorithm 1 Euclidean Distance Transform via IFT.

```
Input: A binary image $I$, a set $S$ of contour pixels in $I$ (seeds), an Euclidean
       adjacency relation $A$, and a labeling function $\lambda(p)$ that assigns a
       segment label to each pixel $p$ in $S$.
Output: The cost map $C$, the root map $R$, and the label map $L$.
Auxiliary data structure: A priority queue $Q$.
begin
    foreach $p \in I$ do
        $C(p) \leftarrow +\infty$; $R(p) \leftarrow NIL$; $L(p) \leftarrow NIL$
    foreach $p \in S$ do
        $C(p) \leftarrow 0$; $R(p) \leftarrow p$; $L(p) \leftarrow \lambda(p)$ insert $p$ in $Q$
    while $Q$ is not empty do
        remove from $Q$ a pixel $p = (x_p, y_p)$ such that $C(p)$ is minimum
        foreach $q = (x_q, y_q)$ such that $q \in A(p)$ and $C(q) > C(p)$ do
            $C' \leftarrow (x_q - x_{R(p)})^2 + (y_q - y_{R(p)})^2$, where $R(p) = (x_{R(p)}, y_{R(p)})$
            is the root pixel of $p$ if $C' < C(q)$ then
                if $C(q) \neq +\infty$ then
                    remove $q$ from $Q$
                $C(q) \leftarrow C'$; $R(q) \leftarrow R(p)$; $L(q) \leftarrow L(p)$ insert $q$ in $Q$
end
```

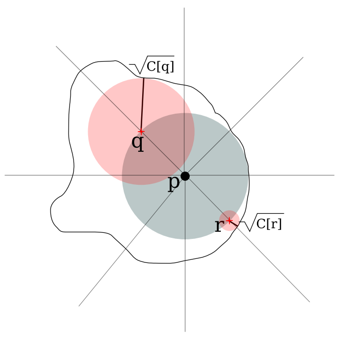
3.2 Faster tensor scale computation for binary images

A considerable speedup in the computation of the tensor scale for binary images is possible by exploiting the following aspect: if we have the shortest distance between a pixel p and the contour, there is no need to search for edge points inside the circle with radius $\sqrt{C(p)}$ (Figure 2(a)).

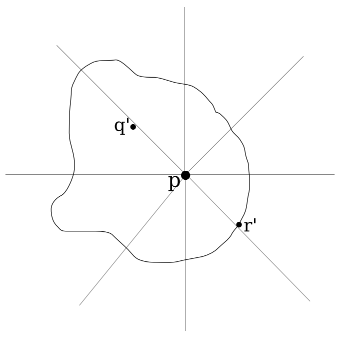
According to Miranda's algorithm, edge points are searched along opposite sample lines, alternately. In our approach, the algorithm jumps along the lines and visits the pixels q and r at the same time (Figure 2(b)).



(a) First step.



(b) Second step.



(c) Third step.

Figure 2. Example of optimization made by using Euclidean IFT.

The searching for edge points continues outside the area defined by the cost $\sqrt{C(p)}$ in Figure 2(b), and the minimum between $\sqrt{C(r)}$ and $\sqrt{C(q)}$ indicates the location for the next jump. These jumps may continue iteratively until the closest edge point along the sample line is found.

In the example, the edge is found at the pixel $R(r)$ (i.e., at the contour point r' nearest to r). The algorithm defines that the two edge points in this sample line are at r' (coordinate of $R(r)$ relative to p) and at q' (coordinate of the point diametrically opposite to r', relative to p), as shown in Figure 2(c).

By performing this procedure for all sample lines, the algorithm defines all edge points and uses the same formula defined by Miranda et al. (Equation 1) for finding the orientation of the ellipse.

The localization of the edge points is formalized in Algorithm 2.

The next section presents orientation mapping based on Euclidean IFT.

Algorithm 2 Edge points localization for ellipse centered at pixel p.

Input: A pixel $p = (x_p, y_p)$, the number m of sample lines, and the cost map C returned by Algorithm 1.

Output: The vector *edge* that contains m pairs of edge points localized at the sample lines.

```
begin
    for $\theta \leftarrow 0°$ to $179°$, with increments $\frac{180}{m}$ do
        $v \leftarrow \sqrt{C(p)}$ $p_1 \leftarrow NIL$; $p_2 \leftarrow NIL$; $q_1 \leftarrow 0$; $q_2 \leftarrow 0$   while $p_1 \neq 0$ and $p_2 \neq 0$ do
            $x \leftarrow v * \cos(\theta)$; $y \leftarrow v * \sin(\theta)$  if $q_1 = 0$ then
                $temp \leftarrow (x_p + x, y_p + y)$; $p_1 \leftarrow \sqrt{C(temp)}$;
            if $q_2 = 0$ then
                $temp \leftarrow (x_p - x, y_p - y)$; $p_2 \leftarrow \sqrt{C(temp)}$;
            $d \leftarrow min(p_1, p_2)$; $v \leftarrow v + d$  $q_1 \leftarrow p_1 - d$; $q_2 \leftarrow p_2 - d$
        $edge[\theta] \leftarrow ((x, y), (x', y'))$, where $(x', y')$ is the coordinate of the point diametrically opposite to $(x, y)$, relative to $p$
end
```

3.3 Feature vector of TSDIZ by orientation mapping onto contour segments

Prior to tensor scale orientation mapping, TSDIZ approach has two stages: tensor scale computation for all pixels inside the object and partition of the object contour into segments.

The orientation mapping uses the label map L returned by the Euclidean IFT (Algorithm 1). The map L groups pixels and their ellipses in the influence zone of each segment. TSDIZ uses as feature vector F the weighted mean of the ellipses orientations in each influence zone of segment. Therefore, F can be indexed by L. The TSDIZ feature vector is formed by the mapped tensor scale orientations (F) and has size equal to the number n_s of segments. Algorithm 3 shows the feature vector computation for TSDIZ.

The function $WeightedAngularMean(V[i])$ returns the weighted angular mean of the ellipses orientations contained in influence zone i, $i = 1, 2, \ldots, n_s$, considering the anisotropies as the weights. The mean $\overline{\theta}$ for angular data [5] is calculated as

$$\overline{\theta} = arctan\left(\frac{\sum_{p=1}^{n} Ani[p] * \sin(2 * Ori[p])}{\sum_{p=1}^{n} Ani[p] * \cos(2 * Ori[p])}\right). \tag{3}$$

Algorithm 3 Feature vector computation for TSDIZ by tensor scale orientation mapping.

Input: A binary image I containing an object O, the number n_s of contour segments, the label map L returned by Algorithm 1, and the vectors Ani and Ori that contain the anisotropies and the orientations of the tensor scale ellipses computed for all pixels of object O, respectively.

Output: A feature vector F that contains the mapped orientation for each contour segment.

Auxiliary data structure: A vector V of n_s lists to store ellipse information in each influence zone.

```
begin
    foreach $p \in O$ do
        insert $(Ani[p], Ori[p])$ in list $V[L(p)]$, where $L(p)$ is the label of the
        influence zone in which $p$ is contained
    foreach $i \in [1, \ldots, n_s]$ do
        $F[i] = WeightedAngularMean(V[i])$
end
```

3.4 TSDIZ similarity function

The similarity function has to determine the rotation difference of the orientations between two TSDIZ vectors. This function also has to determine the position (the segment) in which the feature vectors must be lined up to obtain the best matching between the underlying shapes.

The exhaustive algorithm (Algorithm 4) consists of the registration between the orientation feature vectors. Considering $\alpha = 0°, \ldots, 179°$ and $j = 1, \ldots, n_s$, where n_s is the size of the vectors, the algorithm computes, for each rotation α and for each shift j in the feature vector, the difference between the vectors, after rotating all orientations of one vector by α and circular shifting the same vector by j. The minimun difference obtained corresponds to the distance between the vectors.

In Algorithm 4, the function $AngularDistance(\alpha, \beta)$ gives the smallest angle between the orientations α and β.

The complexity of this algorithm is $O(c * {n_s}^2)$, where c is a constant (in this case, 179). Although it is an exhaustive search, small values of n_s (e.g., $n_s < 70$) makes it still fast. Figure 3 illustrates the registration between two TSDIZ vectors.

4. Experimental results

In this section, the results of the experiments in CBIR are presented.

Algorithm 4 Similarity between two TSDIZ vectors.

```
Input: Two feature vectors $F_A$ and $F_B$.
Output: Distance $dist$ between $F_A$ and $F_B$.
begin
    $dist \leftarrow \infty$ foreach $j \in [1, \ldots, n_s]$ do
        foreach $\alpha \in [0, \ldots, 179]$ do
            foreach $i \in [1, \ldots, n_s]$ do
                $dist_{aux} \leftarrow AngularDistance(\{F_B[(j - i) \mod n_s] + \alpha\}$
                $\mod 180, F_A[i])$ if $dist_{aux} < dist$ then
                    $dist \leftarrow dist_{aux}$
end
```

4.1 Image database

Experiments were conducted using MPEG-7 CE-shape-1 part B [7] database, which consists of 1400 images, categorized in 70 classes (20 images on each class). It is composed by objects silhouettes, like fruits and animals.

4.2 Results

The experiments consist in comparing the TSDIZ and other shape descriptors, with respect to two effectiveness measures used in CBIR – precision versus recall [8] (PR) and multiscale separability [11] (MS separability).

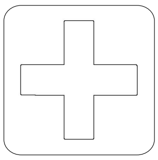
In [11], Torres et al. showed that MS separability represents better than PR curves the separation among clusters (groups of relevant images) in the feature space. This separation is strongly related to the performance of CBIR systems because the search methods rely on them. However, PR is still the most popular effectiveness measure in CBIR. For this reason, we present the results with both measures.

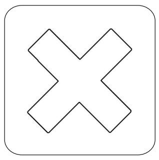
Precision versus recall

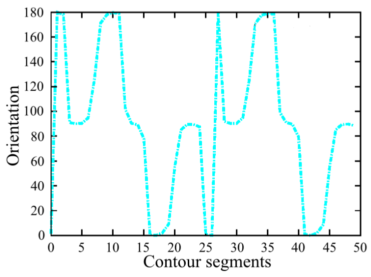
Precision is defined as the fraction of retrieved images that are relevant to the query. In contrast, recall measures the proportion of relevant images among the retrieved images. The precision versus recall curve, or PR curve, indicates the commitment between the two measures and, generally, the highest curve in the graph indicates better effectiveness.

In this experiment, TSDIZ is compared with the following shape descriptors: Beam Angle Statistics [1] (BAS), Multiscale Fractal Dimension [13] (MS Fractal), Moment Invariants [4] (MI), Fourier Descriptor [3] (Fourier), Tensor Scale Descriptor [6] (TSD) and Segment Saliences [11] (SS).

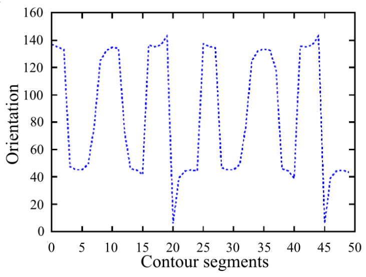
Figure 4(a) presents the PR curves for the evaluated descriptors and TSDIZ with 60 contour segments. The number of segments is a parameter that is tuned for each database and can be learned by training.


(a)

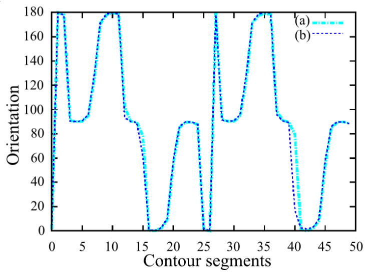

(b)



(c) Orientation curve for (a).



(d) Orientation curve for (b).



(e) Curve matching.

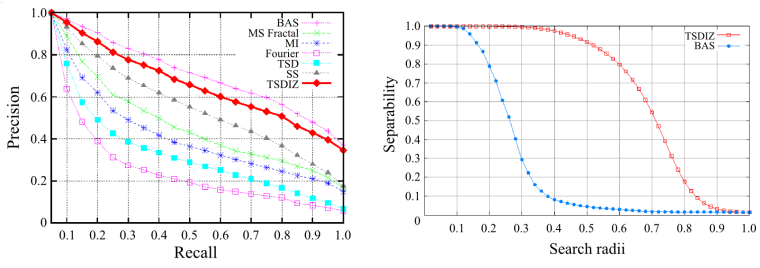
Figure 3. Examples of TSDIZ curves and registration.

TSDIZ descriptor has the second best PR curve among the tested descriptors. BAS descriptor presented the best performance according to PR, as expected for this database [1].

Multiscale separability

A good effectiveness measure should capture the concept of separability. Separability indicates the discriminatory ability between objects that belong to distinct classes. This concept is widely used in cluster analysis, and it was introduced for CBIR by Torres et al. [11].

As TSDIZ has outperformed all other descriptors for MS separability as well, we show in Figure 4(b) the MS separability curves of TSDIZ and BAS only. TSDIZ and BAS present equivalent performance for search radii



(a) PR curves for several descriptors. (b) Multiscale separability curves for TSDIZ and BAS descriptors.

Figure 4. Effectiveness measures experiments conducted in MPEG-7 CE-shape-1 part B database.

less than 10% of their maximum distance. From this point on, the BAS separability curve decreases quickly, indicating that this descriptor is neither robust nor effective for search radii greater than 20%.

Figure 5 shows a visual CBIR example for a query image. The images with a gray background are not in the same class of the query image and should not be returned by the query.

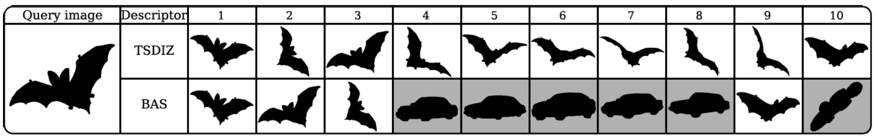


Figure 5. Visual CBIR example.

5. Conclusions and future work

This paper introduced a new shape descriptor based on tensor scale (TS-DIZ). It also provided a faster algorithm for tensor scale computation using Image Foresting Transform (IFT).

The feature vector consists of the tensor scale orientations computed for all pixels of a given object and mapped onto contour segments. The partition of the contour aims at efficiency in encoding contour information and tensor scale orientation mapping aims at storing spatial information into TSDIZ feature vector. These TSDIZ characteristics make the descriptor compact, fast and effective for CBIR.

The experiments done with MPEG-7 CE-shape-1 part B database indicate that TSDIZ has better PR curve than all relevant shape descriptors (except BAS) and the best separability among them, making it the most robust and effective, according to this metric.

The TSDIZ feature extraction function only computes tensor scale ellipses inside objects. Future works will be directed towards incorporating information from ellipses outside the object as well. The TSDIZ descriptor will also be evaluated with other shape databases.

6. Acknowledgments

Authors are grateful to FAEPEX, FAPESP, CAPES, CNPq, and Microsoft for financial support.

References

[1] N. Arica and F. T. Y. Vural, *BAS: a perceptual shape descriptor based on the beam angle statistics*, Pattern Recognition Letters **24** (2003), no. 9–10, 1627–1639.

[2] A. X. Falcão, J. Stolfi, and R. d. A. Lotufo, *The Image Foresting Transform: theory, algorithms, and applications*, IEEE Transactions on Pattern Analysis and Machine Intelligence **26** (2004), no. 1, 19–29.

[3] R. C. Gonzalez and R. E. Woods, *Digital image processing*, 2nd, Addison-Wesley Longman Publishing Co. Inc., 2001.

[4] M.-K. Hu, *Visual pattern recognition by moment invariants*, IEEE Transactions on Information Theory **8** (1962), no. 2, 1962.

[5] K. V. Mardia and P. E. Jupp, *Directional statistics*, John Wiley and Sons, 1999.

[6] P. A. V. Miranda, R. d. S. Torres, and A. X. Falcão, *TSD: a shape descriptor based on a distribution of tensor scale local orientation*, Proceedings of Brazilian Symposium on Computer Graphics and Image Processing, 2005, pp. 139–146.

[7] *The MPEG Project.* <`http://www.chiariglione.org/mpeg`>. Access in: Jan,2007.

[8] H. Müller, W. Müller, D. McG. Squire, S. Marchand-Maillet, and T. Pun, *Performance evaluation in content-based image retrieval: overview and proposals*, Pattern Recognition Letters **22** (2001), no. 5, 593–601.

[9] P. K. Saha, *Tensor scale: a local morphometric parameter with applications to computer vision and image processing*, Computer Vision and Image Understanding **99** (2005), no. 3, 384–413.

[10] R. d. S. Torres and A. X. Falcão, *Content-based image retrieval: theory and applications*, Revista de Informática Teórica e Aplicada **13** (2006), no. 2, 161–185.

[11] ———, *Contour salience descriptors for effective image retrieval and analysis*, Image and Vision Computing **25** (2007), no. 1, 3–13.

[12] R. d. S. Torres, A. X. Falcão, and L. d. F. Costa, *Shape Description by Image Foresting Transform*, Proceedings of International Conference on Digital Signal Processing, 2002, pp. 1089–1092.

[13] ———, *A graph-based approach for multiscale shape analysis*, Pattern Recognition **37** (2004), no. 6, 1163–1174.

[14] D. Zhang and G. Lu, *Review of shape representation and description techniques*, Pattern Recognition **37** (2004), no. 1, 1–19.

On morphological color texture characterization

Erchan Aptoula and Sébastien Lefèvre

UMR-7005-CNRS-LSIIT, Université Louis Pasteur, Illkirch, France
{aptoula,lefevre}@lsiit.u-strasbg.fr

Abstract We investigate the combined use of multiple structuring elements with the standard morphological texture characterization tools, namely morphological covariance and granulometry. The resulting operator is applied to both grayscale and color images in the context of texture classification. As to its extension to color texture data, it is realized by means of a weighting based reduced vector ordering in the IHLS color space, equipped with genetically optimized arguments. The classification experiments based on this framework are carried out with the publically available Outex13 texture database, where the proposed feature extraction scheme outperforms the univariable versions of the operators under consideration.

Keywords: multivariate mathematical morphology, texture, granulometry, covariance, color ordering.

1. Introduction

Mathematical morphology (MM) offers a variety of tools for texture characterization, such as *granulometry*, *morphological covariance*, *orientation maps*, etc. The first two in particular have been employed successfully in a number of texture analysis applications [3, 7, 22, 23].

More precisely, granulometry is a powerful tool based on the “sieving” principle, implemented by means of successive openings and/or closings with structuring elements (SE) of various sizes, hence it is capable of extracting shape and size characteristics from textures. Morphological covariance on the other hand, is based on erosions with pairs of points separated by vectors of various lengths, and provides information on the coarseness, anisotropy as well as periodicity of its input.

In this paper, we concentrate on these two operators, and specifically on the combined exploitation of their SE variables: size, distance and direction. Since the original size-only definition of pattern spectra [13], these operators have been extended in various ways (e.g., color, multivariate, attribute based versions, etc.). Relatively recent applications have explored for instance the combination of SE shape and size as far as granulometry is concerned [24, 25], hence leading to a feature matrix rather than a vector, that describes the combined size and shape distribution of its input. As to covariance, the

G. Banon, et al. (eds),
Mathematical Morphology and its Applications to Signal and Image Processing, 153–164.
©2007 *MCT/INPE. Printed in Brazil.*
URL of this article: http://urlib.net/dpi.inpe.br/ismm@80/2007/03.19.16.53

coupled use of SE pair distance and direction makes it possible to exploit the anisotropic properties of textures additionally to their periodicity [12, 23].

Here we investigate the ways of combining the complementary information extracted by these two operators (e.g., concatenation, dimension reduction, etc.), and propose a hybrid of the two, where SE couples are varied in terms of size, direction as well as distance. The proposed combination scheme is compared in terms of classification accuracy, against the standard definitions, using the publically available Outex13 color texture database. The so far obtained experimental results show that it leads to an improvement over the usual concatenation of feature vectors.

Furthermore, as far as the extension of this operator to color images is concerned, since MM is based on complete lattice theory, a vector ordering mechanism becomes necessary. Hence, we propose a weight based reduced vector ordering, defined on the improved HLS (IHLS) color space, designed specifically for the purpose of color texture classification. This approach makes it possible to optimize, for instance through genetic algorithms, the weight of each component adaptively, according to the training set under consideration.

The rest of the paper is organized as follows. Section 2 introduces briefly granulometry and covariance, and then elaborates on the combination of their variables. In Section 3, the problem of extending morphological operators to multivariate images is discussed, and the proposed ordering is detailed. Next, Section 4 presents the experimental results that have been obtained with the Outex13 database. Finally, Section 5 is devoted to concluding remarks.

2. Morphological texture characterization

In this section, we start by recalling the basic texture categories along with their perceptual characteristics, and then the covariance and granulometry operators are introduced. Moreover, the combination of multiple SE related variables is discussed.

2.1 Texture properties

According to the pioneering taxonomy work of Rao [19], textures can be classified with respect to their spatial distribution of details into the following four categories (Figure 1).

Strongly ordered: Textures consisting of the repetitive placement of their primitive elements according to a particular set of rules.

Weakly ordered: Textures possessing a dominant local orientation, which can however vary at a global level.

Disordered: Textures lacking any repetitiveness and orientation, and usually described on the basis of their roughness.

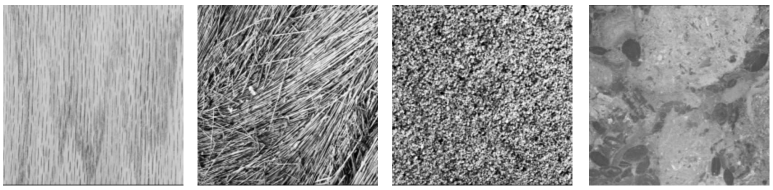


Figure 1. Texture examples from the Brodatz album [5], from left to right, strongly ordered, weakly ordered, disordered and compositional.

Compositional: Textures that do not belong to any of the previous categories, and exhibit a combination of their characteristics.

In an effort to determine efficient features, capable of discriminating among the members of these categories, Rao and Lohse [20] have conducted psycho-physical experiments, and identified *regularity* (or *periodicity*), *directionality* and *complexity* as the most important perceptual texture characteristics, as far as human observers are concerned. With the subsequent work of Chetverikov [6] and Mojsilovic et al. [16], *overall color* and *color purity* were added to this list.

2.2 Morphological covariance

This operator was initially proposed [14, 15, 22] as the equivalent in MM of the autocorrelation operator. The morphological covariance K of an image f, is defined as the volume Vol, i.e., sum of pixel values, of the image, eroded by a pair of points $P_{2,v}$ separated by a vector $\vec{v}$:

$$K(f; P_{2,v}) = \mathrm{Vol}\left(\varepsilon_{P_{2,v}}(f)\right), \tag{1}$$

where ε represents the erosion operator. In practice, K is computed for varying lengths of $\vec{v}$, and most often the normalized version is used for measurements:

$$K^n(f) = \mathrm{Vol}\left(\varepsilon_{P_{2,v}}(f)\right) / \mathrm{Vol}(f). \tag{2}$$

In the light of the aforementioned perceptual properties of textures, given the resulting uni-dimensional covariance series, one can gain insight into the structure of a given image [23]. In particular, the periodic nature of covariance is strongly related to that of its input. Furthermore, the period of periodic textures can easily be determined by the distance between the repeated peaks, that appear at multiples of the sought period; whereas the size of the periodic pattern can be quantified by means of the width of the peaks. In other words, their sharpness is directly proportional to the thinness of the texture patterns appearing in the input. Likewise, the initial slope at the origin provides an indication of the coarseness, with quick drop-off corresponding to coarse textures.

In order to obtain additional information on the directionality of f, one can plot against not only different lengths of $\vec{v}$, but orientations as well [12].

2.3 Granulometry

Granulometry [14, 15] as a term belongs to the field of materials science, where the granularity of materials is determined by passing them through sieves. Using the same principle, this operator consists in studying the amount of image detail removed by applying morphological openings γ_λ and/or closings φ_λ of increasing size λ. The volumes of the opened (or closed) images are then plotted against λ, or more usually their discrete derivative $\mathrm{Vol}(\gamma_\lambda - \gamma_{\lambda+1})$, i.e., *pattern spectrum.* The normalized version of the operator can be written as:

$$G^n(f) = \mathrm{Vol}\left(\gamma_\lambda(f)\right) / \mathrm{Vol}\left(f\right). \tag{3}$$

For unbiased measurements, the volume computation may be reduced only to the area affected by the operator. As a featuring tool, granulometry provides information on the shape and size of ordered textures, and regularity of disordered textures [4, 23]. Furthermore, both operators can be applied on a local or global level.

2.4 Proposed approach

Indeed, considering the fundamental perceptual texture properties mentioned in Section 2.1, morphological covariance and granulometry provide invaluable, yet complementary information on their input. More precisely, covariance extracts a feature vector containing information on periodicity and directionality, whereas granulometry concentrates rather on the granularity of its input. Consequently both are necessary in the general case for an efficient texture description.

However, their combination is rather ambiguous, as it can be realized in a variety of ways. The obvious method, is to calculate independently each feature vector and then employ their concatenation. We propose here an alternative way, which consists in unifying the two operators' functionalities by varying in parallel multiple SE properties. As previously mentioned, the use of multivariate granulometry and covariance has already been reported, specifically, in the form of combined SE direction and distance [9], as far as covariance is concerned, and shape and size combination with granulometry [24, 25].

We choose to implement with this purpose a combination of SE size, direction and distance (Figure 2). For practical purposes, we replace the erosion (ε) operator of covariance (2) with an opening (γ). Of course, on the contrary of granulometry it is also necessary to employ SE pairs, so that periodicity information may be extracted. Hence the following hybrid expression is obtained:

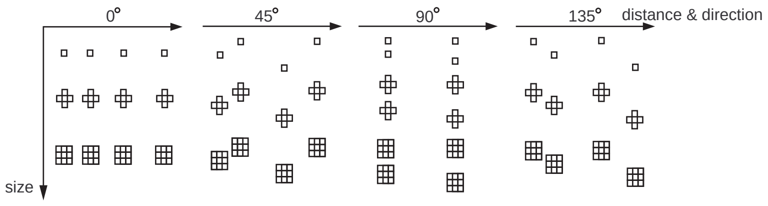


Figure 2. Illustration of structuring element pair variations, with respect to size, direction and distance.

$$GK^n(f) = \mathrm{Vol}\left(\gamma_{P_{\lambda,v}}(f)\right) / \mathrm{Vol}(f)\,, \tag{4}$$

where $P_{\lambda,v}$ denotes a pair of SEs of size λ separated by a vector $\vec{v}$. However, it should be noted that as the sieving principle of multiple morphological openings is satisfied if, and only if the SE is a compact convex set [15], this combination no longer qualifies as a granulometry. In practice, only the four basic directions $(0°, 45°, 90°, 135°)$ are of importance, thus it was chosen to integrate directional variation with distance as shown in Figure 2. Of course, in case directionality becomes particularly significant one can always separate it as an additional dimension representing a finer distinction of directions, or even add one more dimension for shape distributions, where different SE shapes (e.g., disc approximation, square, lines, etc.) are also employed along with direction, size and distance.

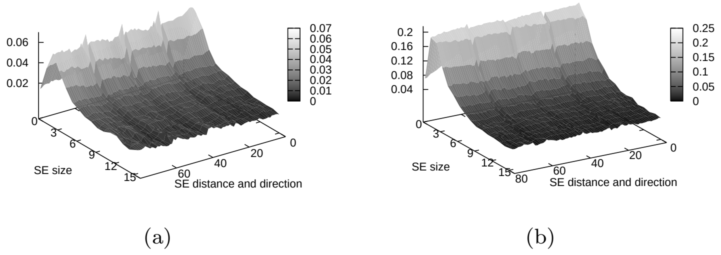


Figure 3. Plot of the feature matrices resulting from the application of expression (4) on the strongly ordered (left) and disordered (right) texture of Figure 1.

Figure 3 presents the plots of the resulting features matrices, as applied to the strongly ordered and disordered textures of Figure 1. Although their size distributions are rather similar, their directionality and periodicitiy are clearly distinct.

Moreover, as far as classification is concerned, feature vector or matrix size is of primary importance, since redundant information may eventually be present and disrupt the overall process. Even with the moderate sizes used in practice, the resulting feature set can easily become excessively large. That is why dimension reduction techniques, such as principal component analysis (PCA), as it will be shown in Section 4, could become necessary. Before testing the proposed approach, as well as the different ways of its use, in the next section a way of extending morphological operators, along with the proposed hybrid operator, to color images is presented.

3. Extending to color images

As previously mentioned, color is an integral part of texture description, and several ways of extracting color features have been reported, e.g., color histograms, color correlograms, etc. According to Palm [18], these techniques can be classified into the following three categories.

Parallel approach: Color and intensity information is processed separately. For instance a color histogram along with a co-occurrence matrix.

Sequential: Color information is first transformed into a greyscale form, which is then processed with the tools available for intensity images.

Integrative: The color channels are processed either separately or simultaneously.

Here we choose to implement the third approach. The extension of morphological operators to color and more generally to multivariate images is still an open problem. Specifically, since the morphological framework is based on complete lattice theory [21], it is theoretically possible to define morphological operators on any type of image data, as long as a complete lattice structure can be introduced on the image intensity range. In other words, at least a partial vector ordering is required. Several approaches have been proposed with this purpose (e.g., marginal ordering, r-orderings, c-orderings, etc.), a comprehensive survey of which can be found in [1].

3.1 Color space

The choice of color space is of fundamental importance, as it can largely influence the end results [11]. Here, we choose to follow the trend of the last years in the domain of color morphology, and employ a polar color space based on the notions of hue ($h \in [0, 2\pi]$), saturation ($s \in [0, 1]$) and luminance ($l \in [0, 1]$). More precisely, although most polar color spaces are essentially just a more intuitive description of the RGB color cube, several implementations exist, e.g., HSV, HSB, HLS, HSI, etc. [8]. According to Hanbury and Serra [10], the cylindrical versions of these spaces have serious inconsistencies and are inappropriate for quantitative color image processing. Hence we make our color space choice in favor of the *improved HLS* space (IHLS) [10] using the L1 norm, which employs the original biconic version of HLS.

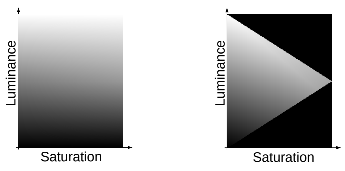


Figure 4. Vertical semi-slice of the cylindrical HLS (left) and bi-conic IHLS (right) color spaces.

As illustrated in Figure 4, one of the most important drawbacks of the cylindrical HLS space is the unintuitive definition of saturation. Specifically, it is possible to have maximized saturation values for zero luminance. This inconvenience, as well as the dependence of saturation on luminance are remedied with the IHLS space, where the maximal allowed value for saturation is limited in relation to luminance. Therefore, in order to benefit from the advantages of polar spaces in the context of multivariate morphology, the ordering of IHLS color vectors is necessary.

3.2 Ordering color vectors

For the sake of simplicity, we have omitted the hue component at this stage of our research from the ordering process, and instead we concentrate on the relations of luminance (l) and saturation (s) (color purity). Luminance is well known to account for the intensity variations and consequently, it is often sufficient for the recognition of most objects, whereas color has a rather auxiliary contribution. These two components may be ordered in a variety of ways, for instance marginally, lexicographically, etc. A marginal ordering strategy in this case:

$$\forall\, \mathbf{c}, \mathbf{c}' \in [0,1]^2\,,\ \ \mathbf{c} \leq \mathbf{c}' \Leftrightarrow c_1 \leq c_1' \ \wedge\ c_2 \leq c_2', \tag{5}$$

is rather inappropriate as it does not take into account inter-channel relations. A lexicographical approach on the other hand, with luminance at top position:

$$\forall\, \mathbf{c}, \mathbf{c}' \in [0,1]^2\,,\ \ \mathbf{c} \leq \mathbf{c}' \Leftrightarrow c_1 \leq c_1' \vee (c_1 = c_1' \wedge c_2 \leq c_2'), \tag{6}$$

prioritizes excessively the first component, since the second dimension (i.e., saturation) does not contribute to the outcome of vector comparisons unless an equality takes place at the first.

We choose to use a reduced or R-ordering where color vectors are first reduced into scalar values and then ranked according to their natural scalar order:

$$\forall\, \mathbf{c}, \mathbf{c}' \in [0,1]^2\,,\ \ \mathbf{c} \leq \mathbf{c}' \Leftrightarrow g(\mathbf{c}) \leq g(\mathbf{c}'). \tag{7}$$

Obviously the main issue at this point is the choice of the scalarization function $g : [0,1]^2 \to \mathbb{R}$. In order to efficiently combine the information contained in saturation and luminance channels, their relations need to be taken into account. Specifically, given the bi-conic form of IHLS, saturation can reach its maximal value for medium luminance levels, whereas it is of minimal importance for extreme levels (i.e., either too dark or too bright). In order to model this relation we choose to use the sigmoid based transition proposed in [2]:

$$l \in [0,1], \quad h(l) = \begin{cases} \frac{1}{1+exp(-k_L(l-l_l))} & \text{if} \quad l \leq 0.5, \\ \frac{1}{1+exp(k_L(l-l_u))} & \text{if} \quad l \geq 0.5, \end{cases} \tag{8}$$

where the slope $k_L = 10$, and the lower and upper offsets are respectively $l_l = 0.25$ and $l_u = 0.75$. Its plot is given in Figure 5. The arguments of $h(l)$ were set empirically, and divide the luminance range roughly in three regions, with the middle corresponding to important saturation levels.

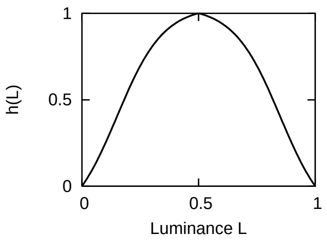


Figure 5. Plot of the weighting function $h(l)$ for the importance of color purity in relation to luminance.

Furthermore, the main problem consists in determining the influence of each component. In other words, in what amount are we to use luminance and saturation when comparing vectors? In the ideal case, one would follow an image or vector specific approach, for example by increasing the contribution of saturation if the image or vectors under consideration are highly saturated. However, this method is in our opinion suitable for intra-image problems such as filtering, but ill suited for inter-image problems, such as texture classification. It results in using different weights for each component depending on the vectors or processed image, hence leading to a highly adaptive approach, which undermines the comparability of the calculated feature sets.

Therefore, we propose to follow a strategy where the contribution of each component is dependent on the image database under consideration. More precisely, g is defined as:

$$w_l, w_s \in \mathbb{R}, \; \forall (l,s) \in [0,1]^2, \; g(l,s) = w_l \times l + w_s \times h(l) \times s, \tag{9}$$

where $w_l + w_s = 1$ are the weights of luminance and saturation respectively. These weights are to be determined by means of a genetic algorithm, or any other means of unsupervised optimization. Specifically, given the training set of textures, the values of w_l and w_s are to be set to their values minimizing the cost function w of features, that in turn minimizes the distance of features among textures of the same class, and maximizes the distance of textures belonging to distinct classes:

$$w = dist_{intra-class} + (1 - dist_{inter-class}). \tag{10}$$

Consequently, the color ordering becomes specific to the image database under consideration. Of course this principle is by no means limited to textures and can be applied in a likewise fashion for optimizing for instance the ordering of multispectral vectors in remote sensing images. Application results are given in the next section.

4. Results

In this section, we present the results that have been obtained using the color textures of Outex13 (Figure 6) [17]. This set consists of 68 textures, where every image has been divided into 20 non-overlapping sub-images, each of size 128 × 128 pixels, thus producing a total of 1360 images, which are evenly divided as training and test sets. We compare four different feature extraction schemes with both grayscale and color images. Specifically, we test features computed using a granulometry (Granulometry) (3), morphological covariance (Covariance) (2), their concatenation (Concatenated) and finally their proposed combined form (Combined).

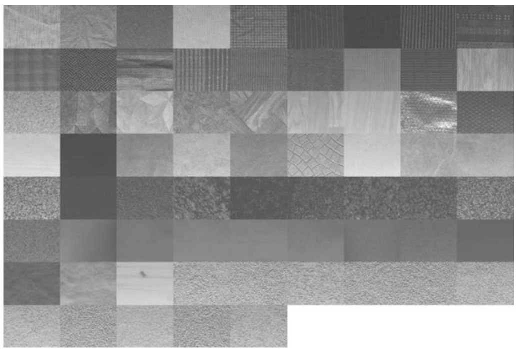


Figure 6. Examples of the 68 textures of Outex13 [11].

More precisely, for granulometry square shaped SEs have been employed, where a SE of size k has a side of $2k + 1$ pixels, and k varied from 1 to 30 in steps of size 2. As to covariance, the four basic directions have been used $(0°, 45°, 90°, 135°)$ in combination with distances varying from 1 to 20. For their concatenated as well as proposed combined form (4) the same arguments were in place. The 80x25 feature matrix that has resulted from the combination option, was reduced into a matrix of size 2x80 by means of a PCA transform and preserving only the first two dimensions. For grayscale computations the luminance component of IHLS has been used, while for the processing of color information the vectorial versions of operators were implemented, based on the ordering (7), the weights of which have been set in two ways. Besides using fixed values (*Color:* $w_l = w_s = 0.5$), the optimization described in Section 3.2 has been also implemented (*Color-optimized*). The image set has been classified using a kNN classifier with $k = 1$ and the Euclidean distance as a similarity metric.

The classification accuracies, computed as the fraction formed by the number of successful classifications divided by the total number of subjects, are given in table Table 1. Globally, one can immediately remark the positive, though comparably to intensity, small effect of using color information. A result which asserts the auxiliary role of color in texture recognition. Moreover, covariance systematically outperforms granulometry, hence showing the higher pertinence of periodicity and directionality with this database, compared to granularity. The combination of the two operators by means of a concatenation improves the accuracy levels, while the proposed hybrid operator provides the overall best results, both with color as well as grayscale images. Additionally, according to the obtained values, the optimization scheme appears to result in database specific feature vectors, hence improving the overall performance.

Table 1. Classification accuracies (%) for the Outex 13 textures.

Features	*Grayscale*	*Color*	*Color optimized*
Granulometry	67.53	68.78	72.03
Covariance	73.82	76.92	80.46
Concatenated	77.75	79.93	83.74
Combined	**83.53**	**85.53**	**88.13**

5. Conclusion

As a fundamental problem of computer vision, several approaches have been developed for texture description. Their extension to color images however is still an open issue. In this paper, we have proposed a method for combining the complementary information provided by the two basic texture featuring tools offered by mathematical morphology. This combination by means of varying multiple SE variables has the advantage of better capturing, with respect to a mere concatenation, the three essential texture properties: periodicity, directionality and complexity. Its extension to color data has been realized using a reduced ordering in the IHLS space. The use of weights results in a flexible solution that makes it possible for each image channel to contribute to the vector comparison outcome. Furthermore, optimization methods may be employed for rendering this ordering data specific.

The experiments that have been carried out on the Outex13 database, have provided indications on the proposed methods' practical interest. Future work will concentrate on the additional exploitation of shape information. Moreover, the use of the hue component holds further potential of improving the efficiency for the computed feature vectors.

References

[1] E. Aptoula and S. Lefèvre, *A comparative study on multivariate mathematical morphology*, Pattern Recognition **40** (November 2007), no. 11, 2914–2929, DOI 10.1016/j.patcog.2007.02.004.

[2] ———, *On the morphological processing of hue*, submitted to Image and Vision Computing (2007).

[3] A. Aubert, D. Jeulin, and R. Hashimoto, *Surface texture classification from morphological transformations*, International Symposium on Mathematical Morphology (Palo Alto, USA, June 16–18, 2000) (J. Goutsias, L. Vincent, and D. S. Bloomberg, eds.), Kluwer Academic Publishers, 2000, Mathematical morphology and its applications to Image and Signal Processing, pp. 253-252.

[4] S. Batman and E. R. Dougherty, *Size Distributions for Multivariate Morphological Granulometries: Texture Classification and Statistical Properties*, Optical Engineering **36** (May 1997), no. 5, 1518–1529.

[5] P. Brodatz, *Textures: a photographic album for artists and designers*, Dover, New York, 1966.

[6] D. Chetverikov, *Fundamental structural features in the visual world*, International Workshop on Fundamental Structural Properties in Image and Pattern Analysis, (Budapest, Hungary, September 6–7, 1999), pp. 47–58.

[7] G. Fricout and D. Jeulin, *Texture classification from their morphological properties*, International Conference on Metrology and Properties of Engineering (Halmstad University, Sweden, September 2003), pp. 42–55.

[8] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison-Wesley, New York, 1992.

[9] A. Hanbury, U. Kandaswamy, and D. A. Adjeroh, *Illumination-invariant Morphological Texture Classification*, International Symposium on Mathematical Morphology (Paris, France, April 18–20, 2005) (C. Ronse, L. Najman, and E. Decencière, eds.), Springer-Verlag, Dordrecht, Netherlands, 2005, Mathematical Morphology: 40 years on, pp. 377–386.

[10] A. Hanbury and J. Serra, *Colour Image Analysis in 3D-polar Coordinates*, International Conference on Image Processing and its Applications (Magdeburg, Germany, September 2003), pp. 124–131.

[11] T. Mäenpää and M. Pietikäinen, *Classification with color and texture: jointly or separately?*, Pattern Recognition **37** (August 2000), no. 8, 1629–1640.

[12] A. Mauricio and C. Figuerido, *Texture analysis of grey-tone images by mathematical morphology: a nondestructive tool for the quantitative assessment of stone decay*, Mathematical Geology **32** (2000), no. 5, 619–642.

[13] P. Maragos, *Pattern spectrum and multiscale shape representation*, IEEE Transactions on Pattern Analysis and Machine Intelligence **11** (1989), 701–715.

[14] G. Matheron, *Eléments pour une Théorie des Milieux Poreux*, Masson, Paris, 1967.

[15] ——, *Random Sets and Integral Geometry*, J. Wiley, New York, 1975.

[16] A. Mojsilović, J. Kovačević, D. Kall, R. J. Safranek, and S. K. Ganapathy, *The vocabulary and grammar of color patterns*, IEEE Transactions on Image Processing **9** (2000), no. 3, 417–431.

[17] T. Ojala, T. Mäenpää, M. Pietikäinen, J. Viertola, J. Kyllönen, and S. Huovinen, *Outex: New framework for empirical evaluation of texture analysis algorithms*, International Conference on Pattern Recognition (Quebec City, Canada, August 2002), pp. 701–706.

[18] C. Palm, *Color texture classification by integrative co-occurrence matrices*, Pattern Recognition **37** (May 2004), no. 5, 965–976.

[19] A. R. Rao, *A Taxonomy for Texture Description and Identification*, Springer-Verlag, New York, 1990.

[20] A. R. Rao and G. L. Lohse, *Identifying high level features of texture perception*, CVGIP: Graphical Models and Image Processing **55** (1989), no. 3, 218–233.

[21] C. Ronse, *Why mathematical morphology needs complete lattices*, Signal Processing **21** (October 1990), no. 2, 129–154.

[22] J. Serra, *Image Analysis and Mathematical Morphology Vol I*, Academic Press, London, 1982.

[23] P. Soille, *Morphological Image Analysis : Principles and Applications*, Springer-Verlag, Berlin, 2003.

[24] E. R. Urbach, N. J. Boersma, and M. H. F. Wilkinson, *Vector-attribute filters*, International Symposium on Mathematical Morphology, (Paris, France, April 18–20, 2005) (C. Ronse, L. Najman, and E. Decencière, eds.), Springer-Verlag, Dordrecht, Netherlands, 2005, Mathematical Morphology: 40 years on, pp. 95–104.

[25] E. R. Urbach, J. B. T. M. Roerdink, and M. H. F. Wilkinson, *Connected Shape-Size Pattern Spectra for Rotation and Scale-Invariant Classification of Gray-Scale Images*, IEEE Transactions on Pattern Analysis and Machine Intelligence **29** (February 2007), no. 2, 272–285.

Micro-viscous morphological operators

Fernand Meyer and Jesús Angulo

Centre de Morphologie Mathématique (CMM), Ecole des Mines de Paris, Fontainebleau, France
`{fernand.meyer,jesus.angulo}@ensmp.fr`

Abstract In most operators of mathematical morphology source and destination are the same: from pixels to pixels. In this paper we present adjunctions where source and destination are not the same. In addition to the pixels of a grid, we also consider the centers of edges linking neighboring pixels. Interesting filters may be constructed using such operators, in particular bi-levelings, where the introduction of some degree of viscosity permits to obtain higher levels of simplifications as with ordinary levelings.

Keywords: adjunction, hexagonal grid, micro-viscous operators, bi-levelings, morphological filtering.

1. Introduction

Connected filters, and in particular levelings have nice and interesting features: they simplify images without blurring the contours. For this reason they are often used as a simplification step before segmentation. Generally, one constructs a strongly simplified version g of the image f to segment, where the contours may be blurred, as is the case after Gaussian filtering, or displaced as is the case for morphological alternate sequential filters. This simplified image is called marker image. The leveling takes as input both functions f and g. It modifies g in order to restore the contours of f, by extending the regional maxima of g under f and the regional minima over f. This extension is obtained by creating flat zones. However, the result is sometimes disappointing, as the reconstruction of f reconstructs far more details as expected, if one takes into consideration the initial degree of simplification of g. For this reason one may want leveling types which reconstructs less details. Various directions have been explored. One consists in extending the regional minima and maxima of g by creating pseudo-flat zones, with a higher extension than strict flat zones [3]. Another consists in introducing some viscosity in the reconstruction process [3, 6]. This last method gives good results, but has the disadvantage to need a large support of information for processing each pixel; this slows down the processing speed and complicates the design of hardware implementations.

In the present paper, we introduce a kind of micro-viscosity in levelings which give good results without extending the window necessary for processing each pixel. It appears that the operators which are needed can be described in terms of adjunctions between the nodes and edges of the raster grid. They exhibit also nice filtering properties for detail simplification or for computing the gradient of noisy images.

2. Levelings and bilevelings

Reminder on levelings

Levelings [1,2] are particular connected operators: they enlarge the existing flat zones and produce new ones [4,5]. A connected operator transforms an image f into an image g in such a way that $\forall\,(p,q)$ neighbors: $g_p \neq g_q \Rightarrow f_p \neq f_q$ (0).

The relation (0) expresses that any contour between the pixels p and q in the destination image g corresponds to a contour in the initial image f at the same place. Levelings are obtained through a specialisation of Relation (0). The basic levelings are characterized by:
$\forall\,(p,q)$ neighbors: $g_p > g_q \Rightarrow f_p \geq g_p$ and $g_q \geq f_q$ (1) meaning that any transition in the destination image g is bracketed by a larger variation in the source image. Other types of transitions between neighboring pixels may be considered. For instance a minimal jump between the pixels p and q may be requested: $g_p > g_q + \lambda \Rightarrow f_p \geq g_p$ and $g_q \geq f_q$, leading to the so called λ-levelings. On the other hand g is a viscosity leveling f iff $g_p < (\gamma g)_q \Rightarrow f_p \leq g_p$ and $(\varphi g)_p < g_q \Rightarrow g_q \leq f_q$. To each type of leveling is associated a type of quasi flat-zone: two pixels x and y belong to the same quasi-flat zone of a function f, if there exists a series of pixels $\{x_0 = x, x_1, x_2, ..., x_n = y\}$ such that there is no transition between g_{x_i} and $g_{x_{i+1}}$. If transitions are of the type $g_p > g_q$, their non existence means $\{g_p \leq g_q$ and $g_p \geq g_q\}$, i.e., $g_p = g_q$. Similarly, the quasi-flatzones of the other leveling types are also obtained by expressing the non existence of transitions between adjacent pixels. For instance the λ-flat zones are characterized by $|g_p - g_q| < \lambda$. If g is a leveling of f, then g is identical to f except in the zones where g is quasi-flat.

The relation $g_p > g_q \Rightarrow g_q \geq f_q$ may be interpreted as $[g_p \leq g_q$ or $g_q \geq f_q]$ $\Leftrightarrow [g_q \geq f_q \wedge g_p]$. As p may be any element of the neighborhood N_q of the central point q, we obtain $g_q \geq f_q \wedge \bigvee\limits_{x \in N_q} g_x$ (2). Since it is always true that $g_q \geq f_q \vee g_q$, Relation (2) is equivalent to $g_q \geq f_q \wedge \left(g_q \vee \bigvee\limits_{x \in N_q} g_x \right) = f_q \wedge \delta g_q$, where δ represents the elementary morphological dilation with a flat structuring element containing the central point and all its neighbors. Taking into account the complete relation (1) yields the following criterion for the basic levelings: $f \wedge \delta g \leq g \leq f \vee \varepsilon g$ (3). Similarly $f \wedge [g \vee (\delta g - \lambda)] \leq g \leq f \vee [g \wedge (\varepsilon g + \lambda)]$ characterizes λ-levelings and $f \wedge \delta\gamma g \leq g \leq f \vee \varepsilon\varphi g$ characterizes viscous levelings (ε is the adjunct erosion of δ, γ and φ are the associated openings and closings).

Starting with any type of simplified version g of the image f, one may transform it into a leveling of f by extending the regional maxima of g under f and the regional minima over f. This extension is obtained by creating flat zones [2]. For instance, in the case of flat levelings, one replaces g by $f \vee \varepsilon g$ for all pixels where $g > f \vee \varepsilon g$ and by $f \wedge \delta g$ for all pixels where $f \wedge \delta g > g$, until the inequalities (3) are everywhere satisfied.

Figure 2 compares the three levelings for the same noisy image; the marker is obtained by an alternate sequential filter of size 4, giving a rough approximation of structures to be preserved. It appears that the amount of simplification is higher for λ and viscous levelings than for flat levelings. In smooth regions, the λ-levelings seem to produce too large flat zones. λ-levelings seem to do a better job for filtering images before segmentation. Their drawback is to necessitate neighborhoods of size 3 for constructing the operators $\delta\gamma$ and φg. Our aim in the next section is to introduce some degree of viscosity into levelings and obtain similar results, but by using only neighborhoods of size 1; like that we accelerate the construction of viscous levelings and ease their hardware implementation.

2.1 Bilevelings

An image g is a bileveling of the image f iff $\forall\,(p,q,s)$ being the summits of an elementary triangle of the hexagonal grid:

$$g_p > g_q \text{ and } g_p > g_s \Rightarrow f_p \geq g_p, \tag{4a}$$

$$g_p < g_q \text{ and } g_p < g_s \Rightarrow f_p \leq g_p. \tag{4b}$$

A morphological characterization

Interesting characterizations may be derived from both relations. As an example consider the implication $[g_p > g_q \text{ and } g_p > g_s \Rightarrow f_p \geq g_p]$. It may be interpreted as $[g_p \leq g_q \text{ or } g_p \leq g_s \text{ or } g_p \leq f_p] \Leftrightarrow [g_p \leq f_p \vee (g_q \vee g_s)]$.

As p and s may be any couple of neighboring pixels of p, we obtain

$$g_p \leq f_p \vee \bigwedge_{(q,s,p)=triangle} (g_q \vee g_s)\,, \tag{5}$$

where $(q,s,p) = triangle$ means that (q,s,p) represent the elementary triangle of the hexagonal grid.

On the other hand, it is always true that $g \leq f \vee g$, which together with Inequality (5) is equivalent with $g_p \leq f_p \vee \left(g_p \wedge \bigwedge_{(q,s,p)=triangle} (g_q \vee g_s) \right)$.

Completing with Relation (4b), we obtain the complete characterisation of bilevelings: g is a bileveling of f iff

$$f_p \wedge \left(g_p \vee \bigvee_{(q,s,p)=triangle} (g_q \wedge g_s) \right) \leq g_p \leq f_p \vee \left(g_p \wedge \bigwedge_{(q,s,p)=triangle} (g_q \vee g_s) \right), \tag{6a}$$

or equivalently

$$f_p \wedge \bigvee_{(q,s,p)=triangle} (g_q \wedge g_s) \leq g_p \leq f_p \vee \bigwedge_{(q,s,p)=triangle} (g_q \vee g_s). \qquad (6b)$$

If g is not a bileveling of f, then the relation (6a) does not hold. So we modify g until this relation becomes satisfied:

- on $\{g_p > f_p\}$, we replace g_p by $f_p \vee \left(g_p \wedge \bigwedge_{(q,s,p)=triangle} (g_q \vee g_s) \right)$,
- on $\{g_p < f_p\}$, we replace g_p by $f_p \wedge \left(g_p \vee \bigvee_{(q,s,p)=triangle} (g_q \wedge g_s) \right)$.

In contrast to ordinary levelings, the relation $\forall (p,q)$ neighbors: $g_p > g_q \Rightarrow f_p \geq g_p$ and $g_q \geq f_q$ is not true. However if $(q,s,p) = triangle$, then $g_p > g_q > g_s \Rightarrow f_p \geq g_p$ and $g_s \geq f_s$. On the other hand, if for the same 3 pixels (q,s,p) forming a triangle we have $f_p > g_p$, $f_s > g_s$ and $f_q > g_q$,then it is not necessarily true that $g_p = g_q = g_s$, but it is granted that the two lowest values are the same.

The operators of Relations (6a) and (6b) will be reinterpret below in terms of adjunctions between the nodes and the edges of the hexagonal grid.

3. A few adjunctions on the hexagonal grid

3.1 Reminder on adjunctions

Let f be a function of Fun($\mathcal{D}$,$\mathcal{T}$) and g be a function of Fun($\mathcal{E}$,$\mathcal{T}$). The two operators α : Fun($\mathcal{D}$,$\mathcal{T}$) $\rightarrow$ Fun($\mathcal{E}$,$\mathcal{T}$) and β : Fun($\mathcal{E}$,$\mathcal{T}$) $\rightarrow$ Fun($\mathcal{D}$,$\mathcal{T}$) form an adjunction if and only if: for any f in Fun($\mathcal{D}$,$\mathcal{T}$) and g in Fun($\mathcal{E}$,$\mathcal{T}$): $\alpha f < g \Leftrightarrow f < \beta g$. Then α is a dilation (it commutes with the supremum of functions in Fun($\mathcal{D}$,$\mathcal{T}$)) and β is an erosion (it commutes with the infimum of functions in Fun($\mathcal{E}$,$\mathcal{T}$)). $\beta\alpha$ is a closing in Fun($\mathcal{D}$,$\mathcal{T}$) and $\alpha\beta$ is an opening in Fun($\mathcal{E}$,$\mathcal{T}$).

3.2 Neighborhood relations on the hexagonal grid

Let us consider a regular hexagonal grid, illustrated in Figure 1. Its basic constitutive elements are the pixels ν appearing as big (red) disks and the edges ν linking adjacent pixels (appearing in bold -green- lines). Given f, a function taking its values on any of these grids, the values taken by f on respectively ν and η are $f(\nu)$ and $f(\eta)$. As we will define a number of operators on these grids, we will consider the elements of the grid itself as operators. The operator ν applied on the function f is the value taken by f on ν : $\nu f = f(\nu)$. Similarly we define $\eta f = f(\eta)$. Let $\overline{\nu}$ be the set of nodes or pixels of the initial grid and $\overline{\eta}$ the set of edges.

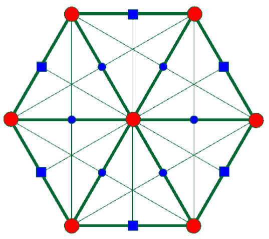


Figure 1. Hexagonal grid: the big disks are the vertices; the other dots are the centres of the vertices.

Supremum, infimum, complementation of these operators are classically defined as (we illustrate the case for $\overline{\eta}$, the definition for $\overline{\nu}$ being similar):

- $[\eta_1 \vee \eta_2](f) = \eta_1(f) \vee \eta_2(f) = f(\eta_1) \vee f(\eta_2)$;
- $[\eta_1 \wedge \vee\eta_2](f) = \eta_1(f) \wedge \eta_2(f) = f(\eta_1) \wedge f(\eta_2)$;
- $-\eta_1(f) = \eta_1(-f)$.

Neighborhood relations Each pixel ν is extremity of 6 edges; in Figure 1, the neighboring edges η of the central pixel appear as small (blue) dots; this neighboring relation is written η/ν, meaning that ν is an extremity of the edge η. Symmetrically, each edge has two extremities; this relation is written ν/η.

Each pixel ν is also summit to 6 adjacent triangles; the edges situated opposite to the central vertex ν are illustrated as (blue) squares. So each node has 6 opposing edges as neighbors; the corresponding neighborhood relation is written $\eta\backslash\nu$. Symmetrically, each edge is common to two adjacent triangles of the grid (consider in Figure 1, one of the edges marked by a small blue dot). So each edge η has as neighbors two opposing summits of triangles. This relation is written $\nu\backslash\eta$.

We now associate to each of these neighborhood relation an erosion and its dual dilation.

Relation between vertices and adjacent edges Let f be a function of Fun($\overline{\nu}$,$\mathcal{T}$) and g be a function of Fun($\overline{\eta}$,$\mathcal{T}$) . The erosion $\varepsilon_{\eta/\nu}$: Fun($\overline{\nu}$,$\mathcal{T}$) $\rightarrow$ Fun($\overline{\eta}$,$\mathcal{T}$) applied to function f is defined by its value at the edge η_i : $\eta_i\varepsilon_{\eta/\nu}f = \bigwedge\limits_{\eta_i/\nu_j} f(\nu_j) = \bigwedge\limits_{\eta_i/\nu_j} \nu_j f$.

Its dual operator, $\delta_{\eta/\nu}$: Fun($\overline{\nu}$,$\mathcal{T}$) $\rightarrow$ Fun($\overline{\eta}$,$\mathcal{T}$) is the dilation: $\eta_i\delta_{\eta/\nu} = \bigvee\limits_{\eta_i/\nu_j} \nu_j$. They are indeed dual as $-\eta_i\delta_{\eta/\nu} = -\bigvee\limits_{\eta_i/\nu_j} \nu_j = \bigwedge\limits_{\eta_i/\nu_j} (-\nu_j)$.
Its adjunct operator maps Fun($\overline{\eta}$,$\mathcal{T}$) into Fun($\overline{\nu}$,$\mathcal{T}$) and uses the symmetrical neighborhood relation ν/η:

$$\nu_j\delta_{\nu/\eta}g = \bigvee_{\nu_j/\eta_i} g(\eta_i) = \bigvee_{\nu_j/\eta_i} \eta_i g.$$

$\nu_j\varepsilon_{\nu/\eta}$ and $\eta_i\delta_{\eta/\nu}$ are indeed adjunct operators as
$\forall i : \eta_i\delta_{\eta/\nu} f < \eta_i g \Leftrightarrow \forall i : \bigvee\limits_{\eta_i/\nu_j} \nu_j f < \eta_i g \Leftrightarrow$
$\forall i, j : \nu_j f < \eta_i g \Leftrightarrow \forall j : \nu_j f < \bigwedge\limits_{\nu_j/\eta i} \eta_i g \Leftrightarrow \forall j : \nu_j f < \nu_j\varepsilon_{\nu/\eta} g.$

These four operators may be summarized in the following table, in which each row represents 2 dual operators and each column two adjunct operators:

$\text{Fun}(\overline{\nu},\mathcal{T}) \to \text{Fun}(\overline{\eta},\mathcal{T})$	$\eta_i\varepsilon_{\eta/\nu} = \bigwedge\limits_{\eta_i/\nu_j} \nu_j$	$\eta_i\delta_{\eta/\nu} = \bigvee\limits_{\eta_i/\nu_j} \nu_j$
$\text{Fun}(\overline{\eta},\mathcal{T}) \to \text{Fun}(\overline{\nu},\mathcal{T})$	$\nu_j\delta_{\nu/\eta} = \bigvee\limits_{\nu_j/\eta_i} \eta_i$	$\nu_j\varepsilon_{\nu/\eta} = \bigwedge\limits_{\nu_j/\eta_i} \eta_i$

In addition we define as previously a dilation and its dual erosion, by taking into account not only the adjacent edges for computing the value at a node but also the node itself:

$\text{Fun}(\overline{\eta} \cup \overline{\nu},\mathcal{T}) \to \text{Fun}(\overline{\nu},\mathcal{T})$	$\nu_j \overbrace{\delta_{\nu/\eta}} = \nu_j \vee \nu_j\delta_{\nu/\eta}$	$\nu_j \overbrace{\varepsilon_{\nu/\eta}} = \nu_j \wedge \nu_j\varepsilon_{\nu/\eta}$

The adjunct operators are defined as follows:

$\text{Fun}(\overline{\nu},\mathcal{T}) \to \text{Fun}(\overline{\eta} \cup \overline{\nu}, \mathcal{T})$	$\nu_j \overbrace{\varepsilon_{\eta/\nu}} = \nu_j$ $\eta_i \overbrace{\varepsilon_{\eta/\nu}} = \eta_i\varepsilon_{\eta/\nu}$	$\nu_j \overbrace{\delta_{\eta/\nu}} = \nu_j$ $\eta_i \overbrace{\delta_{\eta/\nu}} = \eta_i\delta_{\eta/\nu}$

Relation between vertices and opposing edges Similar operators may be based on the neighborhood relations between the nodes and the edges which are in opposition to them, i.e., the neighborhood relations $\eta\backslash\nu$ and $\nu\backslash\eta$. These four operators may be summarized in the following table, in which each row represents 2 dual operators and each column two adjunct operators:

$\text{Fun}(\overline{\nu},\mathcal{T}) \to \text{Fun}(\overline{\eta},\mathcal{T})$	$\eta_i\varepsilon_{\eta\backslash\nu} = \bigwedge\limits_{\eta_i\backslash\nu_j} \nu_j$	$\eta_i\delta_{\eta\backslash\nu} = \bigvee\limits_{\eta_i\backslash\nu_j} \nu_j$
$\text{Fun}(\overline{\eta},\mathcal{T}) \to \text{Fun}(\overline{\nu},\mathcal{T})$	$\nu_j\delta_{\nu\backslash\eta} = \bigvee\limits_{\nu_j\backslash\eta_i} \eta_i$	$\nu_j\varepsilon_{\nu\backslash\eta} = \bigwedge\limits_{\nu_j\backslash\eta_i} \eta_i$

In addition we define as previously a dilation and its dual erosion, by taking into account not only the opposing edges for computing the value at a node but also the node itself:

$\text{Fun}(\overline{\eta} \cup \overline{\nu},\mathcal{T}) \to \text{Fun}(\overline{\nu},\mathcal{T})$	$\nu_j \overbrace{\delta_{\nu\backslash\eta}} = \nu_j \vee \nu_j\delta_{\nu\backslash\eta}$	$\nu_j \overbrace{\varepsilon_{\nu\backslash\eta}} = \nu_j \wedge \nu_j\varepsilon_{\nu\backslash\eta}$

The corresponding adjunct operators are defined as follows:

$\text{Fun}(\overline{\nu},\mathcal{T}) \to \text{Fun}(\overline{\eta} \cup \overline{\nu}, \mathcal{T})$	$\nu_j \overbrace{\varepsilon_{\eta\backslash\nu}} = \nu_j$ $\eta_i \overbrace{\varepsilon_{\eta\backslash\nu}} = \eta_i\varepsilon_{\eta\backslash\nu}$	$\nu_j \overbrace{\delta_{\eta\backslash\nu}} = \nu_j$ $\eta_i \overbrace{\delta_{\eta\backslash\nu}} = \eta_i\delta_{\eta\backslash\nu}$

3.3 Reinterpretation of the micro-viscous operators used in the bilevelings

The four operators used to characterize and to build the bilevelings can be now reinterpreted in terms of adjunctions between the nodes and the edges of the hexagonal grid:

- $\bigwedge_{(q,s,p)=triangle} (g_q \vee g_s) = \bigwedge_{\nu_p \backslash \eta_i} \eta_i \delta_{\eta/\nu} g = \nu_p \varepsilon_{\nu\backslash\eta} \delta_{\eta/\nu} g$;
- $\bigvee_{(q,s,p)=triangle} (g_q \wedge g_s) = \bigvee_{\nu_p \backslash \eta_i} \eta_i \varepsilon_{\eta/\nu} g = \nu_p \delta_{\nu\backslash\eta} \varepsilon_{\eta/\nu} g$;
- $g_p \wedge \bigwedge_{(q,s,p)=triangle} (g_q \vee g_s) = g_p \wedge \bigwedge_{\nu_p \backslash \eta_i} \eta_i \delta_{\eta/\nu} g = \nu_p g \wedge \nu_p \varepsilon_{\nu\backslash\eta} \delta_{\eta/\nu} g = \nu_p \widehat{\varepsilon_{\nu\backslash\eta}} \delta_{\eta/\nu} g$;
- $g_p \vee \bigvee_{(q,s,p)=triangle} (g_q \wedge g_s) = g_p \vee \bigvee_{\nu_p \backslash \eta_i} \eta_i \varepsilon_{\eta/\nu} g = \nu_p g \vee \nu_p \delta_{\nu\backslash\eta} \varepsilon_{\eta/\nu} g = \nu_p \widehat{\delta_{\nu\backslash\eta}} \varepsilon_{\eta/\nu} g$.

Figure 2 presents a detail for a noisy version of the image "Barbara" (Gaussian noise $\sigma = 20$, $SNR(dB) = 20,13$). The marker function for all the examples is an alternate sequential filter of size 4, giving a rough approximation of structures to be preserved. We compare the results of 5 levelings: the standard flat leveling, the lambda leveling ($\lambda = 1$), the viscous leveling (based on $\delta\gamma$ and $\varepsilon\varphi$), a bileveling based on the micro-viscous operators $\widehat{\delta_{\nu\backslash\eta}} \varepsilon_{\eta/\nu}$ and $\widehat{\varepsilon_{\nu\backslash\eta}} \delta_{\eta/\nu}$, and finally a lambda bileveling (defined by the microviscous-operators $g \wedge \big(\varepsilon_{\nu\backslash\eta} \delta_{\eta/\nu} g + \lambda\big)$ and $g \vee \big(\delta_{\nu\backslash\eta} \varepsilon_{\eta/\nu} g - \lambda\big)\big)$. In addition, the corresponding flat zones are given for each leveled image. It is evident that viscous levelings and bilevelings lead to stronger detail simplification and enlargement of (quasi-)flat zones, especially in noisy and texture images like this example. It seems also that the micro-viscous bilevelings preserve the localisation of the original contours better than the viscous leveling. The micro-operations between vertices and edges, and edges and vertices seem interesting to introduce the viscosity in levelings despite their small size.

4. Micro-viscous pseudo-erosions and dilations, derived operators

4.1 Pseudo-erosions and dilations

For an image f of Fun($\overline{\nu}$,$\mathcal{T}$) the hexagonal unitary erosion εf : Fun($\overline{\nu}$,$\mathcal{T}$) $\to$ Fun($\overline{\nu}$,$\mathcal{T}$) (resp. dilation δf) is based on computing the infimum (resp. supremum) of 6 nodes together with the center node in the hexagonal neighborhood, i.e., $\nu_i \varepsilon f = \bigwedge_{\nu_j} f(\nu_j) = \bigwedge_{\nu_j} \nu_j f$. The unitary operation can

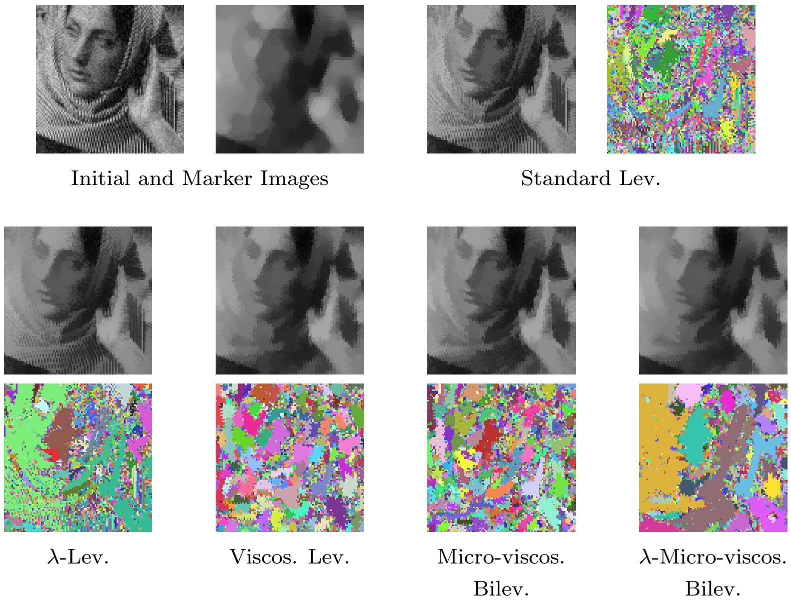


Figure 2. Comparison of standard vs. viscous levelings and bilevelings and their associated λ-flat zones ($\lambda = 1$).

be iterated n-times to build an operator of size n, i.e., $[\varepsilon f]^n$. Similarly to the bilevelings, the micro-viscous operators associated to the adjunctions between the nodes and the edges can be used to introduce other morphological operators.

We propose two uncentered pseudo-erosions for the image f:

- $\xi_{\nu\backslash\eta/\nu} f = \varepsilon_{\nu\backslash\eta}\delta_{\eta/\nu} f$,
- $\xi_{\nu/\eta\backslash\nu} f = \varepsilon_{\nu/\eta}\delta_{\eta\backslash\nu} f$,

where $\xi_{\nu\backslash\eta/\nu}$, $\xi_{\nu/\eta\backslash\nu}$: Fun($\overline{\nu}$,$\mathcal{T}$) $\rightarrow$ Fun($\overline{\nu}$,$\mathcal{T}$). The corresponding uncentered pseudo-dilations $\tau_{\nu\backslash\eta/\nu} f$ and $\tau_{\nu/\eta\backslash\nu} f$ are obtained by taking the dual micro-operators. We denote by $[\xi_{\nu\backslash\eta/\nu} f]^n$ the pseudo-erosion of size n obtained by iteration of n unitary operators. These operators are called pseudo-erosions (resp. pseudo-dilations) in Fun($\overline{\nu}$) because they are increasing (as product of increasing operators) but they do not commute with the infimum (resp. supremum) and the antiextensivity (resp. extensivitiy) in $\overline{\nu}$ is not guaranteed.

Using the micro-viscous operators which take into account the center node during the operation between edges and nodes, it is also possible to define two centered pseudo-erosions:

- $\overbrace{\xi_{\nu\backslash\eta/\nu}} f = \overbrace{\varepsilon_{\nu\backslash\eta}} \delta_{\eta/\nu} f$,

- $\overbrace{\xi_{\nu/\eta\backslash\nu}} f = \overbrace{\varepsilon_{\nu/\eta}} \delta_{\eta\backslash\nu} f$,

and by duality are obtained the two respective centered pseudo-dilations: $\overbrace{\tau_{\nu\backslash\eta/\nu}} f$ and $\overbrace{\tau_{\nu/\eta\backslash\nu}} f$. The centered pseudo-erosions (resp. pseudo-dilations) have the same properties as the uncentered pseudo-erosions (resp. pseudo-dilations) but in addition they are antiextensive (resp. extensive) in $\overline{\nu}$. In addition, both operators $\overbrace{\xi_{\nu\backslash\eta/\nu}} f$ and $\overbrace{\xi_{\nu/\eta\backslash\nu}} f$ are $\geq \varepsilon f$: centered pseudo-erosions are weaker than standard erosions.

4.2 Pseudo-inverses and other evolved operators

To each operator defined above, one may associate its pseudo-inverse operator, obtained by concatenating in reverse order the adjunct operators:

- $\varepsilon_{\nu\backslash\eta}\delta_{\eta/\nu} \dashrightarrow \varepsilon_{\nu/\eta}\delta_{\eta\backslash\nu}$;
- $\delta_{\nu\backslash\eta}\varepsilon_{\eta/\nu} \dashrightarrow \delta_{\nu/\eta}\varepsilon_{\eta\backslash\nu}$;
- $\overbrace{\varepsilon_{\nu\backslash\eta}}\delta_{\eta/\nu} \dashrightarrow \varepsilon_{\nu/\eta}\overbrace{\delta_{\eta\backslash\nu}} = \varepsilon_{\nu/\eta}\delta_{\eta\backslash\nu}$;
- $\overbrace{\delta_{\nu\backslash\eta}}\varepsilon_{\eta/\nu} \dashrightarrow \delta_{\nu/\eta}\overbrace{\varepsilon_{\eta\backslash\nu}} = \delta_{\nu/\eta}\varepsilon_{\eta\backslash\nu}$.

Concatenating such an operator with its pseudo-inverse produces for instance $\varepsilon_{\nu/\eta}\delta_{\eta\backslash\nu}\varepsilon_{\nu\backslash\eta}\delta_{\eta/\nu}$: its construction introduces the opening $\delta_{\eta\backslash\nu}\varepsilon_{\nu\backslash\eta}$ within the closing $\varepsilon_{\nu/\eta}\delta_{\eta/\nu}$.

This operator is increasing, being the product of increasing operators, but it is not a filter as it is not idempotent. However it is an underfilter:

$$\begin{aligned}\varepsilon_{\nu/\eta}\delta_{\eta\backslash\nu}\varepsilon_{\nu\backslash\eta}\delta_{\eta/\nu}\varepsilon_{\nu/\eta}\delta_{\eta\backslash\nu}\varepsilon_{\nu\backslash\eta}\delta_{\eta/\nu} &\leq \varepsilon_{\nu/\eta}\delta_{\eta\backslash\nu}\varepsilon_{\nu\backslash\eta}\delta_{\eta\backslash\nu}\varepsilon_{\nu\backslash\eta}\delta_{\eta/\nu}\\ &= \varepsilon_{\nu/\eta}\delta_{\eta\backslash\nu}\varepsilon_{\nu\backslash\eta}\delta_{\eta/\nu},\end{aligned}$$

since $\delta_{\eta/\nu}\varepsilon_{\nu/\eta}$ is antiextensive and $\delta_{\eta\backslash\nu}\varepsilon_{\nu\backslash\eta}$ is idempotent.

Similarly $\overbrace{\varepsilon_{\nu\backslash\eta}}\delta_{\eta/\nu}\varepsilon_{\nu/\eta}\delta_{\eta\backslash\nu}$ is an underfilter whereas $\delta_{\nu\backslash\eta}\varepsilon_{\eta/\nu}\delta_{\nu/\eta}\varepsilon_{\eta\backslash\nu}$ and $\overbrace{\varepsilon_{\nu\backslash\eta}}\delta_{\eta/\nu}\varepsilon_{\nu/\eta}\delta_{\eta\backslash\nu}$ are overfilters.

Note that other operators can be defined as a product of unit pseudo-erosions/dilations. For instance, a pseudo-opening can be defined as

$$\tau_{\nu\backslash\eta/\nu}\xi_{\nu\backslash\eta/\nu} = \delta_{\nu\backslash\eta}\varepsilon_{\eta/\nu}\varepsilon_{\nu\backslash\eta}\delta_{\eta/\nu},$$

the corresponding pseudo-closing is obtained as $\xi_{\nu\backslash\eta/\nu}\tau_{\nu\backslash\eta/\nu}$, and *mutatis mutandis* other pseudo-openings and closings are obtained with the other unitary micro-operations. Then, the product of pseudo-openings and closings leads to more evolved operators such as the pseudo-alternate sequential filters (pseudo-ASF). A detailed study of the properties of these operators is out of the scope of this paper but we would like to show a few examples. In

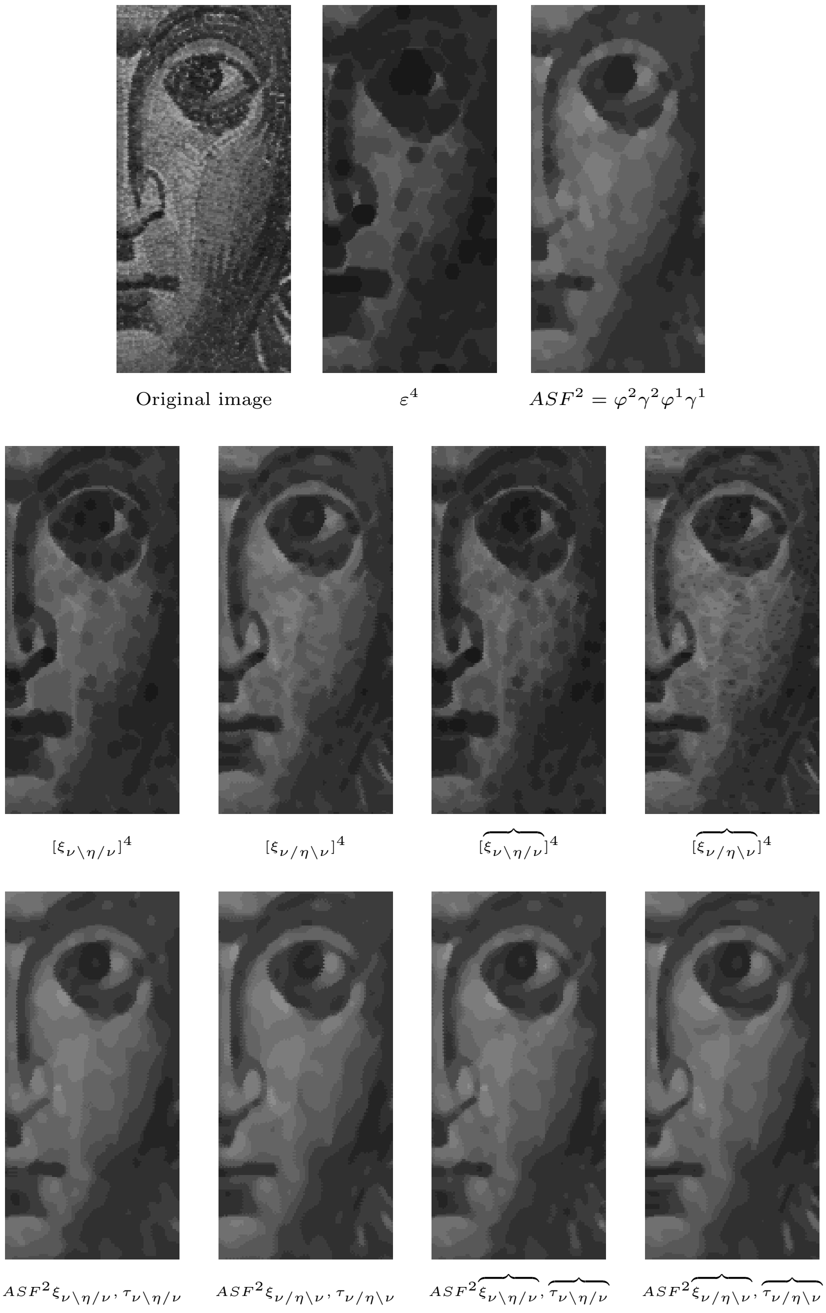
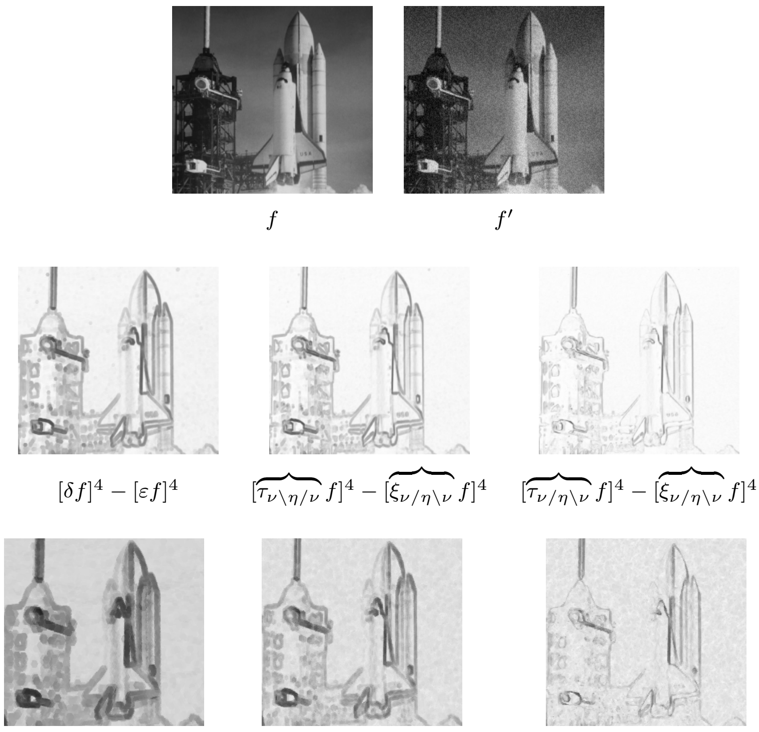

Original image
ε^4
$ASF^2 = \varphi^2\gamma^2\varphi^1\gamma^1$
$[\xi_{\nu\backslash\eta/\nu}]^4$
$[\xi_{\nu/\eta\backslash\nu}]^4$
$[\overbrace{\xi_{\nu\backslash\eta/\nu}}]^4$
$[\overbrace{\xi_{\nu/\eta\backslash\nu}}]^4$
$ASF^2\xi_{\nu\backslash\eta/\nu}, \tau_{\nu\backslash\eta/\nu}$
$ASF^2\xi_{\nu/\eta\backslash\nu}, \tau_{\nu/\eta\backslash\nu}$
$ASF^2\overbrace{\xi_{\nu\backslash\eta/\nu}}, \overbrace{\tau_{\nu\backslash\eta/\nu}}$
$ASF^2\overbrace{\xi_{\nu/\eta\backslash\nu}}, \overbrace{\tau_{\nu/\eta\backslash\nu}}$

Figure 3. Comparison of micro-viscous operators: uncentered and centered pseudo-erosions of size 4 (second row) and uncentered and centered pseudo-alternate sequential filters of size 2 (third row) using a detail of the image "Greek Mosaic" (the original image, the standard hexagonal erosion of size 4 and the standard hexagonal alternate sequential filter of size 4, are given in the first row).

Figure 3 is given a comparison of the different micro-viscous pseudo-erosions and pseudo-alternate sequential filters presented above on a detail of image “Greek Mosaic”. These results of filtering are compared with corresponding standard hexagonal operators. Both kinds of micro-viscous operators $\xi_{\nu\backslash\eta/\nu}$ and $\xi_{\nu/\eta\backslash\nu}$, and the derived pseudo-ASF, have more selective effects than the standard hexagonal counterpart operators, and they result in a regularization of contours. The effects of operators starting from opposing edges η/ν are stronger than starting from adjacent edges $\eta\backslash\nu$: this is due to the distance between the pair of nodes used to compute the edge value. Moreover, note that the uncentered pseudo-erosions (and the derived operators) which are nor antiextensive neither extensive, perform very well for detail simplification; in contrast the centered pseudo-erosions propagate the isolated dark details.



f f'

$[\delta f]^4 - [\varepsilon f]^4$ $[\overbrace{\tau_{\nu\backslash\eta/\nu}} f]^4 - [\overbrace{\xi_{\nu/\eta\backslash\nu}} f]^4$ $[\overbrace{\tau_{\nu/\eta\backslash\nu}} f]^4 - [\overbrace{\xi_{\nu/\eta\backslash\nu}} f]^4$

$[\delta f']^4 - [\varepsilon f']^4$ $[\overbrace{\tau_{\nu\backslash\eta/\nu}} f']^4 - [\overbrace{\xi_{\nu/\eta\backslash\nu}} f']^4$ $[\overbrace{\tau_{\nu/\eta\backslash\nu}} f']^4 - [\overbrace{\xi_{\nu/\eta\backslash\nu}} f']^4$

Figure 4. Comparison of standard vs. micro-viscous centered thick-gradient of size 4 (negative of gradients are shown): Top, “Shuttle” original and corrupted image (gaussian noise $\sigma = 20$, $SNR(dB) = 19,37$); middle, thick gradients for original image and down, thick gradients for noisy image.

The properties of micro-viscous pseudo-erosions/dilations make them interesting for instance to define gradients which are robust to noise. Figure 4 depicts a comparison of standard vs. micro-viscous centered thick-gradient of size 4 for the image "Shuttle" and its corrupted version with gaussian noise. The thick-gradient is defined as the difference between the (pseudo-) dilation and the (pseudo-)erosion of size 4. As we can observe, the gradients based on micro-viscous operators are much more robust to noise than the standard ones. Moreover, the thickness of contours depends strongly on the chosen couple of operations between the nodes and the edges.

5. Conclusions and perspectives

The hexagonal grid offers the highest degree of rotational symmetry and a dense packing of pixels. The implementation of microviscous operators on this grid is simple and elegant. However, it is not complicated to implement similar operators on the square grid, and more generally on weighted graphs: it is sufficient to define erosions and dilations between nodes, adjacent edges and adjacent faces.

The extensions of this work will be in three directions: (i) explore more completely all adjunctions between sub-elements of the hexagonal and square grid, (ii) extend the method to various grids in 3D, (iii) extend the method to arbitrary weighted neighborhood graphs.

References

[1] G. Matheron, *Les nivellements*, Centre de Morphologie Mathématique, Ecole des Mines de Paris, Internal Note N-07/97/MM., Februry 1997.

[2] F. Meyer, *From connected operators to levelings*, ISMM'98 (1998), Mathematical Morphology and its Applications to Image and Signal Processing (Heijmans and Roerdink, Eds.), pp. 191–199.

[3] ______, *The levelings*, ISMM'98 (1998), Mathematical Morphology and its Applications to Image and Signal Processing (Heijmans and Roerdink, Eds.), pp. 199–206.

[4] P. Salembier and J. Serra, *Flat zones filtering, connected operators and filters by reconstruction*, IEEE Trans. on Image Processing **3** (1995), 1153–1160.

[5] J. Serra and P. Salembier, *Connected operators and pyramids*, (1993), Proceedings of SPIE, San Diego, Vol. 2030, pp. 65–76.

[6] I. Terol-Villalobos and D. Vargas-Vazquez, *A Study of Openings and Closings with Reconstruction Criteria*, (2002), Proceedings of ISMM'02 (Talbot and Beare, Eds.), pp. 413–423.

Scene text localization based on the ultimate opening

THOMAS RETORNAZ and BEATRIZ MARCOTEGUI

Centre de Morphologie Mathématique (CMM), Ecole des Mines de Paris, Fontainebleau, France
{retornaz,marcotegui}@cmm.ensmp.fr

Abstract Database indexing and retrieval tools can enormously benefit from the automatic detection and processing of textual data in images. We present a new connected-component based method using morphological numerical residues for automatic localization of text in general image database. This technique tends to be robust to font, scale and slant changes and detects superimposed as well as scene text. The robustness of our approach is proven by the results in *ImagEval* evaluation campaign, which database included old postcards, graphic schemes, stamps, indoor and outdoor scene images and also images without any textual data. In spite of the wide variety of texts and images our approach obtains interesting results without parameter tuning for each image class.

Keywords: scene-text localization, connected component approach, ultimate opening, ImagEval, indexing.

1. Introduction

Multimedia databases, both personal and professional, are developing at a high rate and the need for automated management tools is now imperative. The effort devoted by the research community to content-based image indexing is also huge, but bridging the semantic gap is difficult: the low level descriptors used for indexing (e.g., interest points, texture descriptors) are not enough for an ergonomic manipulation of big and generic image databases. The text present in a scene is usually linked to the semantic context of the image and constitutes a relevant descriptor for content-based image indexing.

Many algorithms focusing on scene text detection have been designed in the past few years. The reader may refer to [8] and [10] for a complete survey of text detection applications and systems. Basically, text localization approaches can be sorted into two main categories. The first category is texture based (which includes derivative and frequency approaches). Jung [7] use as features the pixel values in a star-like pattern. Clark [3] propose five localized features and combine them to get candidate text regions, and claim that their approach is invariant to scale and orientation.

G. Banon, et al. (eds),
Mathematical Morphology and its Applications to Signal and Image Processing, 177–188.
©2007 *MCT/INPE. Printed in Brazil.*
URL of this article: http://urlib.net/dpi.inpe.br/ismm@80/2007/03.19.21.38

The frequency domain is also used: Fourier transform [15], discrete cosine transform [4], wavelet [17], multi-resolution edge detector [2]. These methods perform quite well for small characters, because small texts produce strong texture response. Their extension to generic text imposes important constraints on character alignment and/or color distribution together with the introduction of a multiscale strategy. The second category is the connected component (CC) approach. Some authors use color quantization process [16], morphological operations [5] or split-and-merge algorithm [11] to obtain candidate CCs. These methods could effectively deal with different types of text but require too many heuristic rules such as aspect ratio of characters, horizontal constraint.

Our work focuses on a very generic image database, as illustrated in Figure 7. It contains a wide variety of texts in terms of size, color, font (even manuscript), complex background and also typical scene text deformations due for example to perspective or non planar support. We will build our system on the three following hypotheses: the text should be readable (contrasted), it should be composed of group of characters, and characters of the same text zone should have similar geometric features.

Our system consists in 5 steps that will be described in corresponding sections hereafter. In Section 2, the non linear scale-space approach based on morphological numerical residues is presented. It allows us to extract candidate CCs using contrast information. Section 3 introduces a two-step filter which removes CCs that are certainly non text, detected on basic and easily computed features. Section 4 gives features for discriminating text CCs from non text ones. Then the Section 5 presents the learning strategy. Finally an alignment and merging analysis to filter out the last remaining non-text CCs is introduced in Section 6. Section 7 describes evaluation issues and overall results in the *ImagEval* evaluation campaign. Section 8 is devoted to our conclusions and perspectives.

2. Ultimate opening

Beucher [1] has recently introduced a non linear scale-space based on morphological numerical residues. It enables us to select CCs by trying to avoid at least a priori on size information. Here is an overview of ultimate openings.

2.1 Reminder on attribute opening and closing

Binary connected opening We define a 2D binary image as a subset of $\mathbb{Z}^2$. A binary image X can be decomposed into its connected components $C_i \in \mathcal{C}$, where $\mathcal{C}$ is a connectivity class and i some index set I. We can extract a connected component CC to which belongs an element $x \in X$ by using a binary connected opening Γ_x (see [14] for theoretical background).

Criterion and trivial opening A trivial opening Γ_κ uses an increasing criterion κ to accept or reject a connected set (that κ is increasing means A satisfies κ which implies B satisfies κ for all $B \supseteq A$). Usually κ is in the form

$$\kappa(CC) = (AttributeValue(CC) \geq \lambda), \tag{1}$$

with AttributeValue(CC) some real-valued attribute of CCs (such as area, height,...) and λ the attribute threshold. The trivial opening Γ_κ on a CC simply means that if $\kappa(CC) = True$, we keep CC otherwise we discard it.

Binary attribute opening A binary attribute opening in X consists in a trivial opening of each CC. We can define it as follows:

$$\Gamma^\kappa(X) = \bigcup_{x \in X} \Gamma_\kappa(\Gamma_x(X)). \tag{2}$$

Gray-scale case To apply the definitions to gray scale case we can use the basic threshold decomposition process. Fast implementations can be found in [13].

The top-hat associated with attribute opening or closing is a powerful operator to select image structures but a priori knowledge of size of these structures is required. Next we show the interest of some residual operators, in particular ultimate opening/closing.

2.2 Definition of ultimate opening/closing

The ultimate opening θ was introduced by Beucher [1]. This residual operator analyses the evolution of each pixel x of a grayscale image I subject to a family of openings of increasing sizes (with size parameter λ). The difference between two consecutive openings (named residue) is considered and two significant pieces of information are kept for each pixel. $R_\theta(I)$ registers the value of the maximal residue (contrast information) and $q_\theta(I)$ registers the size of the opening that produced the maximal residue (i.e., the size of the structure that contains the considered pixel). Note that this maximum may not be unique. In that case, as proposed by Beucher, we keep the greatest λ value for which this maximum occurs.

The ultimate opening is then defined as follows:

$$\begin{array}{l} R_\theta(I) = sup(r_\lambda(I)), \forall \lambda \geq 1 : \text{ with } r_\lambda(I) = \gamma_\lambda - \gamma_{\lambda+1}, \\ q_\theta(x) = max(\lambda) : \lambda \geq 1, r_\lambda(x) = R_\theta(x) \text{ and } > 0. \end{array} \tag{3}$$

Now we denote ν^γ an ultimate opening linked with a family of morphological openings and υ^γ the ultimate closing defined by duality.

Figure 1 illustrates the action of υ^γ on a profile, here we have used closings with a family of linear structuring elements of increasing size.

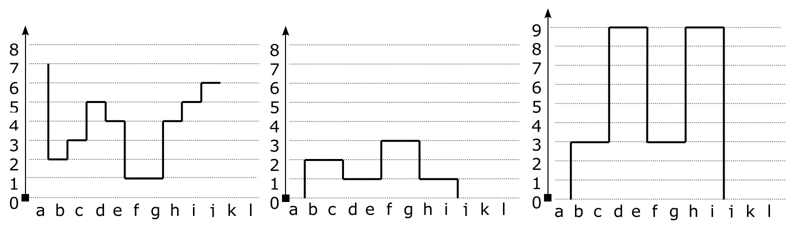


Figure 1. Operator υ^{γ}. From left to right: profile I, R_{θ} and q_{θ}.

2.3 Ultimate opening/closing and attribute

The definitions of ν^{γ}/ υ^{γ} using morphological opening (respectively closing) could be extended by using attribute opening (respectively closing). We denote ν^{κ}/ υ^{κ} these operators and the new definitions of R_{θ} and q_{θ} are obtained replacing γ by Γ^{κ} in Equation 3. We can re-use the Figure 1 to illustrate the action of υ^{κ}, we just replace morphological closings by area closings of increasing size.

2.4 Extracting CC using ultimate attribute closing

Different attributes may be considered for the ultimate opening. For text detection, the height of the bounding box containing CCs is the most suited one. The largest opening considered is λ equal to 1/3 of the image height because 1) characters are rarely larger and 2) we avoid artifacts that usually occur at the late steps of the ultimate opening.

The CCs are computed by thresholding the residual image R_{θ}. First a global low threshold (value fixed at 8) is applied in order to remove really low contrasted structures. Then, a local adaptive threshold is applied on each CC. The aim is to separate CCs possibly merged due to the extremely low threshold previously used. This local threshold is based on the mode of R_{θ} in each CC. Since ultimate opening tends to give the same R_{θ} valuation to all the pixels of a contrasted structure, it makes sense to use the R_{θ} mode. A threshold of mode/2 is applied in each CC. See Figure 2 for an example.

Note that only dark CCs have been extracted. In order to detect also light CCs, the same procedure should be applied to the inverse of the image.

Finally, note also that we only use luma information. ν may be applied to some "color gradient" in order to partially integrate colorimetric information. Due too a significant number of very thin letters in the database (particularly in the old postcard subset), we decide to stop going further.

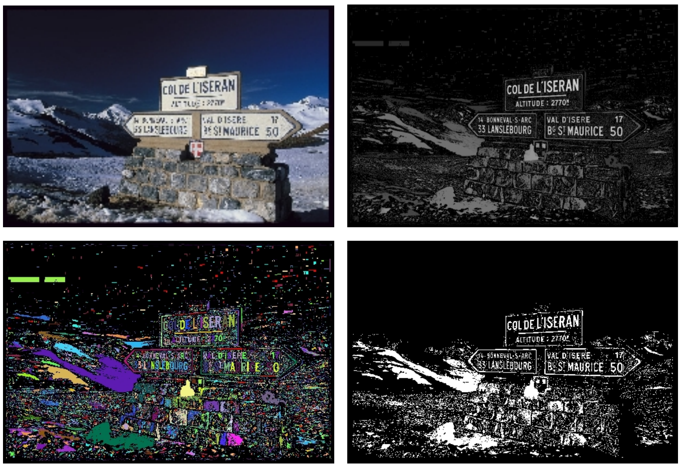


Figure 2. CCs extraction: From left to right and up to down: Original Image I, $R_\theta(I)$ (Gamma Corrected $\gamma = 3$) and $q_\theta(I)$ (randomized), detected CCs.

3. Coarse filtering

In the previous step a lot of CCs were obtained. Most of them do not correspond to characters and can be removed by simple tests as those proposed in this section.

Textured regions typically produce a lot of CCs. They deserve particular attention or a lot of false positives may appear. Therefore we observed the answer of texture to ultimate opening and developed a simple module for texture removal. This module consists in counting the number of CCs grouped by a unitary dilation of the initial CC set. If this number is high (used value 500), the grouped CCs are removed.

A second simple consideration also leads to remove a great number of CCs. Characters are supposed to be made of strokes of constant thickness. We estimate the line thickness of each CC and we remove any CC whose height is smaller than two times its thickness. The thickness is estimated using the distance function in horizontal and vertical directions.

Figure 3 shows that most of non text CCs were successfully discarded at the end of the two-step process.

4. Features to discriminate characters

The number of CCs was considerably reduced in the previous step. We will now characterize the remaining ones in more details. We will classify components in text/non text categories based on the following features.

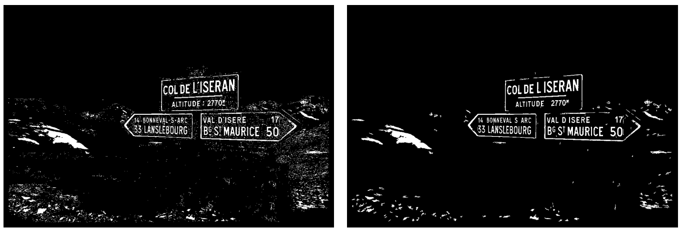


Figure 3. Coarse filtering. Left: result after removing grouped CCs. Right: result after the second step.

Geometric features The simplest features we can consider for a CC is the height (H_{bb}) and width (W_{bb}) of its bounding box, the area of the CC (A_{cc}), and the area of the bounding box (A_{bb}). The inverse values of (H_{bb}), (W_{bb}) and (A_{cc}) are also introduced, allowing us to compute aspect ratios. Some systems commonly use these features and their combinations to discard non text CCs (especially for superimposed characters). In the context of scene text their characterizing performances fall down. We will combine these features, which assess the scale of the character, with more discriminant features. 7 geometrical features are considered.

Stroke thickness estimation A character is supposed to be composed of lines of constant thickness. This seems to be a key characteristic for text CC discrimination. However its estimation is not so easy, this is why several estimators are considered. We propose first to compute the max of distance function inside CCs (Max_{cc}), which generally leads to thickness overestimation because of stroke intersections. We also use run-length distribution inside CCs (as Ashida System in [12]), more precisely we compute the distributions (line per line and column per column) of run-length inside CC. Several parameters are selected from these distributions: mean, median and variance of the RLE distribution (RLE_{MedX}, RLE_{MedY}, RLE_{MoyX}, RLE_{MoyY}, RLE_{VarX}, RLE_{VarY}). RLE is generally sensitive to text deformations.

Moreover, we consider the coherence of the stroke thickness. We compute the distributions of differences (between consecutive lines or columns) of run-length inside CCs. We keep the mean and variance of these distributions (ΔRLE_{MoyX}, ΔRLE_{MoyY}, ΔRLE_{VarX}, ΔRLE_{VarY}). 11 features estimating the stroke thickness and consistency are used altogether.

Shape regularity features Text CCs have more regular shapes than arbitrary CCs. We propose to compute some shape parameters which express this regularity. For example text CCs have a limited number of holes, rather regular contours and important compactness. So to assess these character-

Table 1. Confusion table of our learning system.

GT	*Detected*	
	Characters	*No Characters*
Characters	89.1	10.9
No Characters	9.7	90.3

istics, we propose the following features: Euler Number E, perimeter P, compacity (A_{cc}/P^2) and complexity (P/A_{cc}) (as [9]). The last three characteristics are also computed after filling the holes of CCs because we think that this transformation could be relevant to discriminate text from non text CCs (which makes 7 shape regularity features).

Contrast We estimate the contrast of a CC as the Maximum Inter Class Variance ($M.V.I$) in its bounding box. We include this feature because the text is supposed to be contrasted in order to be readable. We also include ($M.V.I$/Overall Variance) which is a normalized feature.

5. Machine learning

We use machine learning techniques in order to achieve the classification of text/non text CCs base on the 27 features presented above. To this purpose we use quadratic Linear Discriminant Analysis (LDA) [6], which attempts to predict the classification as a linear combination of the selected features and cross products.

The classifier was trained on 177000 CCs extracted from 350 images and labeled by hand as text or no text. The classical cross-validation technique has been used to compute the cost function and to evaluate the performances of the designed classifier. The confusion table is shown in Table 1. A misclassification rate of about 10 % is obtained for both text and no text CCs.

Figure 4 presents typical classification results of the designed classifier. CCs of Figure 3 (right) classified as text are shown in Figure 4 (top). Figure 4 (down) shows the result of coarse filtering (left) and the CCs classified as text leading to false positives in the church image (see Figure 7).

6. Grouping characters in text zones

Text zones are rarely composed of a single character. This remark is often used to reduce the number of false positives at the cost of a few false negatives. A commonly accepted constraint is to impose at least 3 characters in order to accept a text zone. We develop a two step alignment and merging analysis.

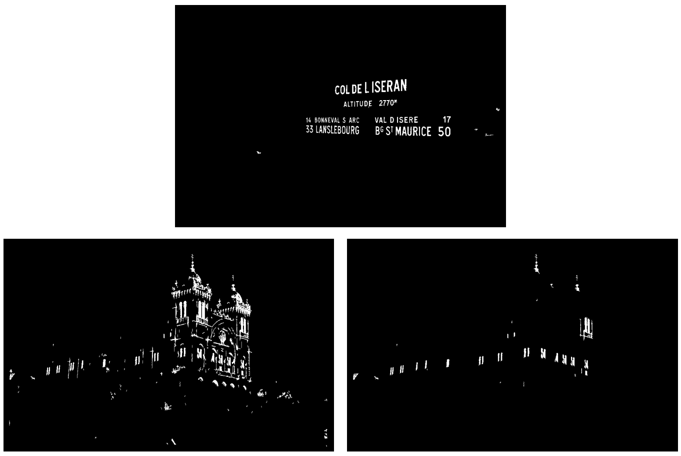


Figure 4. Machine Learning. Top: CCs of Figure 3 (right) classified as text. Bottom left: result of the coarse filtering. Right: CCs classified as text.

Strong layout We start from the CCs classified as text by the learning stage and we merge CCs if they verify the following constraints (see Figure 5):

- the difference of corresponding bounding boxes heights is smaller than the smallest bounding box height ($|H_1 - H_2| < min(H_1, H_2)$);
- the distance of the bounding boxes centers in the vertical direction is less than 70% of the smallest bounding box height ($\Delta_{Cy} < 0.7min(H_1, H_2)$);
- the distance of the bounding boxes centers in the horizontal direction is less than the smallest bounding box height ($\Delta X < min(H_1, H_2)$).

Only groups of at least 3 characters are preserved.

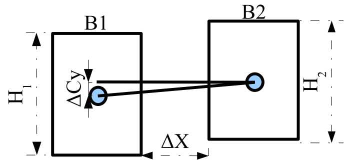


Figure 5. Parameters used in the merging process.

Relaxation of merging strategy The text zones detected in the previous step are considered as seeds and extended in the horizontal direction (see Figure 6 (left)) by a certain amount (here 200 pixels). A new merging process is applied in these extended areas with the following constraints: 1) the merging criteria are the same that those used for seed creation, 2) groups of at least 2 CCs are accepted and 3) CCs classified as no text by the learning approach take part in the process. If groups of at least 2 CCs are found, they are added to the final detected box (see the '17' string in Figure 6 (right)). Thus groups of only 2 CCs can be detected, but only if they are in the neighborhood of a seed. Since CCs classified as no text are considered, some misclassification of the learning step are recovered.

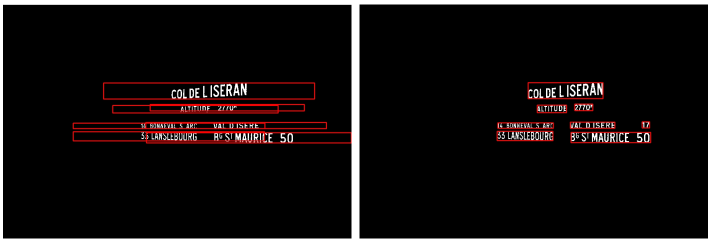


Figure 6. Grouping strategy. Left: resulting seeds after the application of the strong layout with extended boxes for the relaxation step. Right: result after relaxation step.

Final Grouping The *ImagEval* committee established some rules in order to define the ground truth (see `www.imageval.org` for details). We merge all the boxes verifying the merging criterion defined by the committee. Finally, given that we detect clear and dark text separately, we merge boxes if they intersect.

7. Experimental results

7.1 Evaluation issues

The global database contains 500 images including old postcards, graphic schemes, stamps, indoor and outdoor scene images and also images without any textual data. Furthermore images can be grayscale or color. The ground truth database was created by the evaluation committee and was not known by the participants. The committee used the ICDAR[1] metric for performance evaluation (see [12]). This metric is based on the intersection ratio between detected text zones and ground truth. If no text is detected

[1]International Conference on Document Analysis and Recognition

for an image without textual information, both precision and recall are set to 1. The results are given in Section 7.3.

7.2 Parameters of the system

Along the paper several parameters have been introduced. Given that the ground truth was unknown, systematic optimization was not possible. All these parameters have been empirically tuned in order to cope with the huge variability of the blind test database. A second database with similar characteristics has been provided for the official test. The robustness of our method is proven by the results we have obtained (see section below).

Note that we provide simply an overall result. In fact it is difficult to quantify the effects of each steps separately because our approach finds CCs, then the ground truth is known at the bounding box level. Even if we have CCs tagged as text, there is still a gap (known as the character restoration step) before we could submit our CCs to an Optical Character Recognition system.

7.3 Overall results

The results were given to the participants by the *ImagEval* evaluation committee. The database contains 500 images (190 old postcards, 206 color images and 104 black and white images). Table 2 gives the precision, recall and F-mean computed with the ICDAR performance metric [12].

Table 2. Overall Performance.

Precision	Recall	F-Mean
0.48	0.65	0.55

Table 3. Overall Performance ICDAR dataset.

Precision	Recall	F-Mean
0.41	0.57	0.48

In Figure 7 the variability of the database and the visual quality of the results can be observed. In general the results are satisfactory and the proposed method seems to cope with the variability of text zones in scale (Figures 7(e), 7(h)), font (Figure 7(d)), and slant (Figure 7(f)). Some false positives are detected, for example in structured zones (such as barriers or balcony handrails). Some false negatives are also present: the merging rules do not take into account vertical text and therefore it is discarded. If the characters are not correctly defined, as it is the case in the postal card, the CC extraction step fails.

We also provide in Table 3, results from *free* ICDAR database[2]. Note that we merge results on TrailTrain and TrialTest subset. All system param-

[2]Aviable at `http://algoval.essex.ac.uk/icdar/Datasets.html` sec: Robust Reading and Text Locating

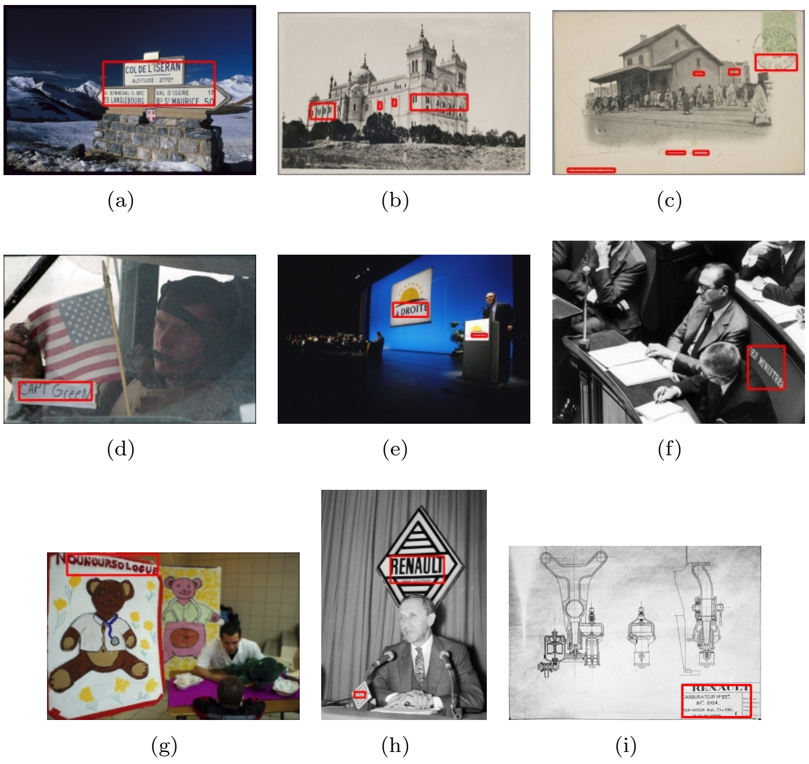


Figure 7. Examples of detected text zones.

eters remain, except for the last merging step (i.e., **Final Grouping**). We replace the merging criterion defined by *ImagEval* committee by a simple inclusion criterion due to a huge difference on annotation strategy (*ImagEval* committee annotates at the sentence level whereas ICDAR committee annotates at the word level). No additional efforts were made to improve the results.

8. Conclusions

The whole system has been empirically tuned for the extremely diverse *ImagEval* database without ground truth information. In spite of the wide variety of text and images, a promising score is obtained for the whole database. The ground truth is now available, and we are considering parameter optimization in a probabilistic framework. The strong point of our system is the relative robustness against changes in scale, fonts, complex backgrounds and typical deformations of scene text (perspective, non planar support). The system may be significantly improved for a restricted application by modifying specifically the learning step. In future work we plan to integrate the q_θ (the second piece of information provided by the ultimate opening operator), an important source of information not used yet, in the CC extraction step.

References

[1] S. Beucher, *Numerical residues*, Mathematical morphology: 40 years on, 2005, pp. 23–32.

[2] X. Chen, J. Yang, J. Zhang, and A. Waibel, *Automatic detection and recognition of signs from natural scenes*, IEEE Transactions on Image Processing **13** (2004January), no. 1, 87–99.

[3] P. Clark and M. Mirmehdi, *Recognising text in real scenes*, International Journal on Document Analysis and Recognition **4** (2002August), no. 4, 243–257.

[4] D. Crandall, S. Antani, and R. Kasturi, *Extraction of special effects caption text events from digital video*, International Journal of Document Analysis and Recognition **5** (2003April), no. 2-3, 138–157.

[5] Y. M. Y. Hasan and L. J. Karam, *Morphological text extraction from images*, IEEE Trans. Image Processing **9** (2000November), no. 11, 1978–1983.

[6] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning : Data mining, inference, and prediction*, 1st ed. 2001. Corr. 3rd printing, 2003, Springer Series in Statistics, Springer, New York, 2001.

[7] K. Jung and J. Han, *Hybrid approach to efficient text extraction in complex color images*, Pattern Recognition Letter **25** (2004April), no. 6, 679–699.

[8] K. Jung, K. I. Kim, and A. K. Jain, *Text information extraction in images and video: a survey*, Pattern Recognition **37** (2004May), no. 5, 977–997.

[9] M. León, S. Mallo, and A. Gasull, *A tree structured-based caption text detection approach*, Proc. fifth iasted international conference visualization, imaging and image processing, 2005September, pp. 220–225.

[10] J. Liang, D. Doermann, and H. Li, *Camera-based analysis of text and documents: a survey*, International Journal on Document Analysis and Recognition **7** (2005July), no. 2-3, 84–104.

[11] R. Lienhart and W. Effelsberg, *Automatic text segmentation and text recognition for video indexing*, Multimedia Syst. **8** (2000), no. 1, 69–81.

[12] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, R. Young, K. Ashida, H. Nagai, M. Okamoto, H. Yamamoto, H. Miyao, J. Zhu, W. Ou, C. Wolf, J. M. Jolion, L. Todoran, M. Worring, and X. Lin, *Icdar 2003 robust reading competitions:entries, results, and future directions*, International Journal on Document Analysis and Recognition **7** (2005July), no. 2-3, 105–122.

[13] A. Meijster and M. H. F. Wilkinson, *A comparison of algorithms for connected set openings and closings*, IEEE Transactions on Pattern Analysis and Machine Intelligence **24** (2002April), 484–494.

[14] J. Serra, *Image analysis and mathematical morphology: Theorical advances*, Vol. 2, Academic Press, 24-28 OvalRoad, London NW1 7DX, 1988.

[15] B. K. Sin, S. K. Kim, and B. J. Cho, *Locating characters in scene images using frequency features*, International conference on pattern recognition, 2002, pp. III: 489–492.

[16] K. Wang and J. A. Kangas, *Character location in scene images from digital camera*, Pattern Recognition **36** (2003October), no. 10, 2287–2299.

[17] Q. Ye, Q. Huang, W. Gao, and D. Zhao, *Fast and robust text detection in images and video frames*, Image and Vision Computing **23** (2005June), no. 6, 565–576.

News from viscousland

Corinne Vachier[1] and Fernand Meyer[2]

[1] *Centre de mathématiques et de leurs applications (CMLA), Ecole Normale Supérieure de Cachan, France*
`Corinne.Vachier@cmla.ens-cachan.fr`

[2] *Centre de Morphologie Mathématique (CMM), Ecole des Mines de Paris, Fontainebleau, France*
`Fernand.Meyer@cmm.ensmp.fr`

Abstract Viscous closings have been presented at ISMM'02 as an efficient tool for regularizing the watershed lines in gray-scale images. We consider now the problem of reconnecting several edge portions of a same object. In the binary case, this is very nicely solved via the computation of the distance function to the grains: the downstream of the saddle points reconnects the grains, and is known as the perceptual graph. As a particular case, overlapping particles may be separated by computing the watershed line of the inverse distance function. This paper extends the approach to grey-tone images using the concept of viscous dilations. Finally, combinations of both viscous dilations and viscous closings are proposed for segmenting objects with dotted and irregular contours.

Keywords: morphological filtering, viscous closing, viscous dilation, crest line.

1. Introduction

Crest lines of numerical functions play a fundamental role in many image analysis applications. In segmentation for example crest lines of gradient images coincide with shapes edges. And there are many other situations where crest lines are meaningful: road detection in air images, vessel extraction in medical images or writing analysis... (see Figure 1).

There are two major difficulties when trying to detect crest lines in gray-tone images. First, crest lines are generally not iso-level lines: the luminance varies, the lines are dotted; a prior reconnection of the lines sections is required before engaging their extraction. Second, when images are noisy or fuzzy, crest lines are often irregular and a geometrical regularization is necessary.

The problem of regularizing thin crest lines in gray-tone images has been addressed in previous works [4–6, 8] and has led to the powerful concept of viscous transformation.

What is the fundamental idea of the viscous transformations ? Basically, morphological filtering is based on openings or closings. The shapes in images are simplified according to a structuring element having a predefined

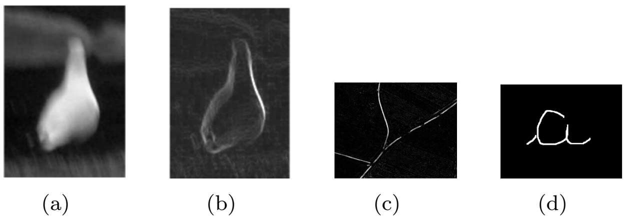


Figure 1. Some applications where crest lines have to be detected on gray-tone images. (a) Original gray-scale image. (b) its gradient. Crest lines of the gradient image correspond to the shapes edges. (c) A problem of road detection in a gray-tone air image. (d) A hand writing example (in a binary image).

shape and size. The filtering activity increases with the size of the structuring element. In classical gray-tone morphology, the same transformation is applied on all level sets of the function.

For different reasons, it is often necessary to adapt the regularization to the local luminance information available in the images: structures of high luminance are supposed to be perfectly known while structures of low luminance require a higher amount of modeling. The idea of the viscous transformations is to combine the effects of a whole family of closings (or openings) of decreasing activity in such a way that low luminance areas are severely smoothed whereas points of high luminance are left unchanged.

Two different combinations where proposed in [8] inspired by the behavior of viscous fluids. The first mimics the propagation by an oil type fluid, the second by a mercury type fluid. It has been proven in [8] that the two models are equivalent for functions which are cylinders, i.e., functions made of thin crest lines (of one pixel thick) and of null points.

In [5] viscous closings have been extended to any family of increasing operators of decreasing activity, as for example families of dilations. The present paper will show how dotted thin crest lines in gray-tone images may be reconnected and at the same time smoothed using viscous transformations. As we will see, this idea is not completely new.

The questions we mentioned are completely classical in the binary case. However, their resolution in the case of numerical functions raises some difficulties. A binary example is presented in Figure 2. By closing, irregular portions of the curve are enlarged while linear ones are left unchanged. A smoothed and thin version of the original curve is very easily obtained by extracting the median axis of the closed set. In this example, the median axis is computed by successive morphological thinning. Of course, the structuring element used must be homotopic [1, 7].

Independently of the smoothing, the question of the connection of the different curve portions is very easily solved by the use of the distance function.

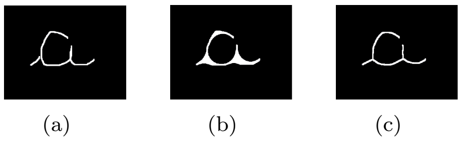


Figure 2. Regularization of a thin line by closing. (a) Original binary image. (b) Result of a closing by a disk. (c) Extraction of the median axis by homotopic thinning.

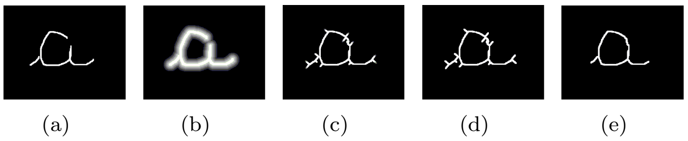


Figure 3. (a) Original binary image. (b) Distance map computed on a narrow band. (c) Homotopic numerical thinning of the distance map. (d) Crest lines of the thinned image. (e) Crest lines remaining after pruning.

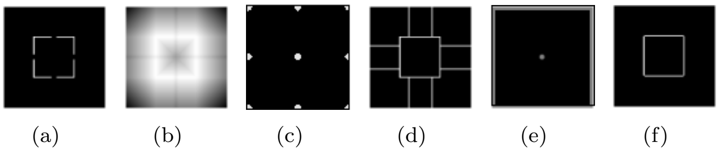


Figure 4. Standard morphological algorithm for connecting binary contours. (a) Dotted square. (b) Inverse distance function. (c) Regional minima of the negative distance function. (d) Watershed line of the inverse distance function. (e) Imposed markers. (f) Watershed line associated with the predefined markers.

Let X represents a thin binary structure made of several connected components and suppose that we want to join the components of X in order to produce a unique thin connected set while minimizing the length of the final structure. The standard method in the binary case consists in two steps:

- first, the computation of the distance of any point x of the space to the set X. This step issue is the formation of a distance map f_d:

$$f_d(x) = d(x, X) = \inf\{d(x, y),\, y \in X\}. \qquad (1)$$

  Of course, $f_d(x) = 0$ if x belongs to X. Note that the connected components of X are the regional minima of f_d.

- second, the extraction of the minimal paths in f_d connecting the set X. Or equivalently, detecting the saddle points and their downstream, yielding the so-called perceptual graph [3], which can also be done by extracting the most significant crest lines of the inverted distance function. As a particular case, overlapping particles may be separated by computing the watershed line of the inverted distance function.

This very classical algorithm is illustrated in Figures 3 and 4. In practice, only interesting crest lines are selected. The watershed transform associated with a predefined set of markers is one of the most elegant solution for this task (see Figure 4). In the example presented in Figure 3 however, the standard algorithm has been modified: the distance map is computed in a narrow band (of width 20 pixels); it is then thinned by an homotopic numerical thinning [1, 7]. The regional minima being eliminated, the final set results from a morphological pruning of the crest lines.

The purpose of the work presented in this paper is to extend the procedure developed for binary sets to gray-tone images. This will be possible thanks to the viscous dilations. The general framework of viscous transformations being recalled in the next section, viscous dilations are then introduced and their properties studied. Several examples are then presented for illustrating the pertinence of the proposed method.

2. Viscous transforms

Viscous transformations were firstly introduced for regularizing the watershed transform but their applications field largely exceeds the strict segmentation framework. What we discuss here was essentially already developed in [8]. The current presentation is however notably different. In [8], only viscous closings were studied; in the present paper, the concept is enlarged to any increasing operator as suggested in [5].

Let us consider a family of morphological operators of increasing activity. By morphological, we mean increasing operators. Let $(T_n)_{n=0:N-1}$ denotes the family. T_n being increasing, it preserves the order between sets or functions: $f < g \Rightarrow T_n(f) < T_n(g)$.

T_n being of increasing activity means: if $n > p$, $T_p(f)$ is closer to f than $T_n(f)$. If, in addition, the T_n are supposed to be extensive, this leads to: $Id = T_0 < T_p < T_n$

Well known examples of such families are the granulometries (by opening or by closing) or the pyramid of dilations or erosions (associated with homothetic convex structuring elements). In order to simplify the presentation, we restrict ourselves to extensive operators (closings or dilations for example); the case of anti-extensive operators is dual; the case of auto-dual operators is more delicate and will not be treated here.

Rather than computing each filter resultant one by one as is the case in granulometric analysis, the idea of the viscous transformations is to combine the effects of the whole family of filters in a unique formulation. The image points are not anymore identically processed. The filtering parameter n is adapted to the local luminance so that regions with low luminance are strongly smoothed whereas regions of high luminance are left unchanged.

$$\begin{tikzcd}
f \arrow[r, "\text{decomposition}"'] \arrow[d, phantom, "\|"] & \{\chi_h(f)\}_{h\geq 0} \arrow[d, "\text{Id}"] \\
\bigvee_{h\geq 0} h.\chi_h(f) & \{\chi_h(f)\}_{h\geq 0} \arrow[l, "\text{reconstruction}"]
\end{tikzcd}$$

Figure 5. Level set decomposition and reconstruction of an upper semi-continuous function

$$\begin{tikzcd}
f \arrow[r, "\text{decomposition}"'] & \{\chi_h(f)\}_{h\geq 0} \arrow[d, "T_{N-h}"] \\
\bigvee_{h\geq 0} h.T_{N-h}[\chi_h(f)] \arrow[u, "T_N^v"] & \{T_{N-h}[\chi_h(f)]\}_{h\geq 0} \arrow[l, "\text{reconstruction}"]
\end{tikzcd}$$

Figure 6. The oil model viscous transformation: each level set is processed independently

The first solution consists in examining the function level set by level set and in indexing the filtering parameter n to the level h: $n = N - h$ for example.

Let $X_h(f)$ and $\chi_h(f)$ denote the level set of the function f at level h and the associated indicatrix function:

$$X_h(f) = \{x \in E,\ f(x) \geq h\} \text{ and } \chi_h(f) = \begin{cases} 1 & \text{if } x \in X_h(f) \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

The function f is supposed to be positive so $h \geq 0$, and of course upper semi-continuous so $\forall h \geq 0$, $X_{h+1}(f) \subset X_h(f)$, and

$$f = \bigvee_{h\geq 0} h.\chi_h(f), \tag{3}$$

which means that f can be processed level set by level set if the process is increasing (see Figure 5). As example, T_n being supposed to be increasing, it satisfies:

$$T_n(f) = \bigvee_{h\geq 0} h.T_n[\chi_h(f)]. \tag{4}$$

If now n depends of h ($n = N - h$), the level set decomposition leads to the following definition (see Figure 6):

$$T_N^v(f) = \bigvee_{h\geq 0} h.T_{N-h}(\chi_h(f)). \tag{5}$$

T_N^v corresponds to the *oil model* described in [8]. For illustration, let us consider a thin line presented in Figure 7. T_n being extensive, the thin

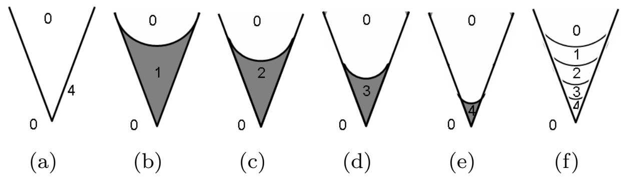


Figure 7. Viscous transformation (oil type). T_n correspond to closings; (a) f is a thin angular line. (b)$1.T_{N-1}[\chi_1(f)]$ (c) $2.T_{N-2}[\chi_2(f)]$ (d) $3.T_{N-3}[\chi_3(f)]$ (e) $4.T_{N-4}[\chi_4(f)]$ (f) $T(f) = \bigvee_{h \geq 0} h.T_{N-h}[\chi_h(f)]$.

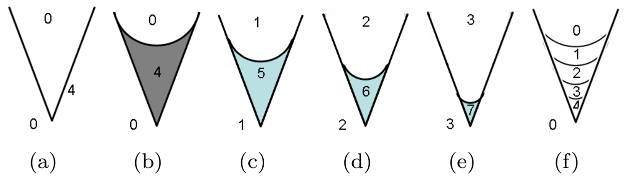


Figure 8. Viscous transformation (mercury type). (a) f (b) $T_N[f]$ (c) $T_{N-1}[f+1]$ (d) $T_{N-2}[f+2]$ (e) $T_{N-3}[f+3]$ (f) $T(f) = \bigwedge_{h \geq 0} T_{N-h}[f+h]$.

line is enlarged by T_n; the cone's interior is smoothed by decreasing size openings: it grows as a viscous lake does, if one interprets the altitude h as a temperature and the filtering parameter $(N-h)$ as a viscosity indicator.

Rather than indexing the filtering parameter n on the luminance h, it can be more interesting to index it on the contrast. The reasoning leading to the second viscous transformation model is detailed in [8]. It is inspired from the behaviour of a *mercury type of fluid*:

$$\tilde{T}_N^v(f) = \bigwedge_{h \geq 0} T_{N-h}(f+h).$$

The mercury type viscous transformation behavior is illustrated on Figures 8 and 9. In both oil and mercury cases, the results of filters T_n when n varies, are stacked. It has been proved in [8] that these two models are equivalent in the case where the function is a set of cylinders on top of a background of value 0.

3. Viscous openings and closings

As said previously, operators T_n may be any extensive morphological transformation whose activity decreases with n and notably closings. As example, viscous closings are defined as follow:

$$\varphi_{R_0}^v(f) = \bigvee_{h \geq 0} h.\varphi_{R_0-h}(\chi_h(f)) \text{ and } \tilde{\varphi}_{R_0}^v(f) = \bigwedge_{k \geq 0} \varphi_{R_0-k}(f+k), \qquad (6)$$

where $R_0 - h$ is the size of the structuring element.

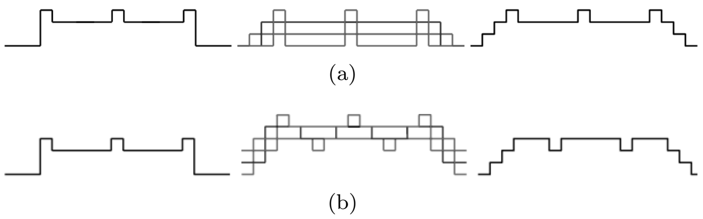


Figure 9. Difference between oil and mercury type of viscous dilation. (a) In the first case (the oil type), details of high luminance are preserved. (b) In the second type (the mercury type), the filtering activity is function of the contrast and not of the luminance. Details of low contrast are dilated.

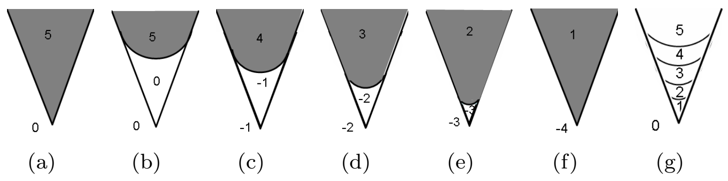


Figure 10. Viscous opening. (a) Original set. (b-f) Results of openings by disk of decreasing size. (g) Result of the viscous opening. Details of the original shape are associated with low levels and coarse descriptions to high levels.

The viscous opening's expressions are derived by duality. However, the formulation is simpler in the mercury case than in the oil case. In the mercury case, it expresses as follow:

$$\tilde{\Gamma}^v_{R_0}(f) = \bigvee_{k \geq 0} \gamma_{R_0 - k}(f - k). \tag{7}$$

Let us now examine what kind of images are build via viscous openings. The Figures 10 and 11 illustrate viscous openings behavior. In the case of the set presented in Figure 10, oil and mercury models are equivalent. The example of Figure 11 illustrates the openings granulometric property [2]: details and coarse shapes are represented at opposite granulometric scales. The viscous transform gathers the entire granulometric information in a unique formulation where coarse sets and details are stacked: due to the size-brightness correlation, details of shapes of low luminance are lost; the luminance of the smallest shapes is lowered. Viscous closings have a dual effect on crest lines as illustrated in Figures 12 and 13. After a viscous or non viscous closing, a thin line is very simply restored by thinning.

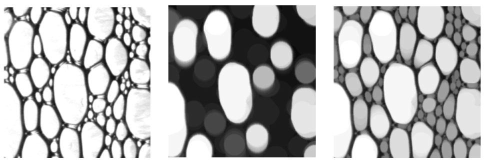


Figure 11. Effect of the viscous opening. (a) Original image of bubbles. (b) Result of a standard opening by a Euclidian disk of size 20. (c) Result of a viscous opening of size 20. Here, the mercury model is chosen since the image is made of catchment basins located at different altitudes.

4. The viscous dilations

We arrive now at the heart of the paper: the relation between distance maps and viscous transforms and their use for reconnecting structures in gray-tone images.

Let us consider a binary set X in a space E. We want to compute the Euclidean distance from any point $x \in E$ to X. When restricting ourselves to the discrete case, the distance map (also called distance function) can easily be computed via erosions by disks of increasing size. And conversely, the negative distance map is computed via dilations of increasing size. Let δ_n denote the dilation by an Euclidean disk of radius n. Any point belonging to the dilated set $\delta_n(X)$ is at a distance lower or equal to n from X:

$$x \in \delta_n(X) \Leftrightarrow d(x, X) \leq n. \tag{8}$$

Considering functions rather than sets:

$$\delta_n(\chi)(x) = 1 \Leftrightarrow d(x, X) \leq n, \tag{9}$$

where $\forall x \in E$, $\chi(x) = 1$ if $x \in X$ and $\chi(x) = 0$ otherwise. χ is nothing but the numerical function of value 0 or 1 defined on the space E and associated with the binary set X.

The computation of the distance map in a narrow band of size $N-1$ around X involves the partial sum:

$$\sum_{n=0}^{N-1} \delta_n(\chi) \text{ with } \sum_{n=0}^{N-1} \delta_n(\chi)(x) = \begin{cases} N & \text{if } x \in X, \\ N-n & \text{if } n-1 < d(x,X) \leq n, \\ 0 & \text{if } d(x,X) > N-1. \end{cases} \tag{10}$$

The points of the set X form the crest lines of the negative distance function. Moreover the operator $\sum_{n=0}^{N-1} \delta_n(\chi)$ is nothing but a viscous dilation. Indeed, the sets $\delta_n(X)$ are nested: $X = \delta_0(X) \subset ... \subset \delta_n(X) \subset ... \subset \delta_{N-1}(X)$. Points belonging to $\delta_n(X) \setminus \delta_{n-1}(X)$ are at distance n from X.

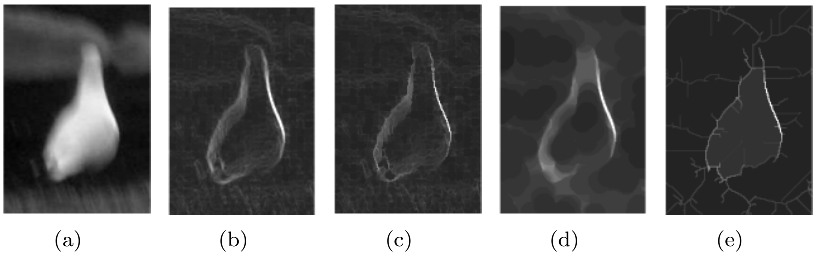


Figure 12. (a) Original image. (b) Its morphological gradient. (c) The homotopic thinning of the gradient. (d) Standard closing of the gradient. (e) Homotopic thinning of (d)

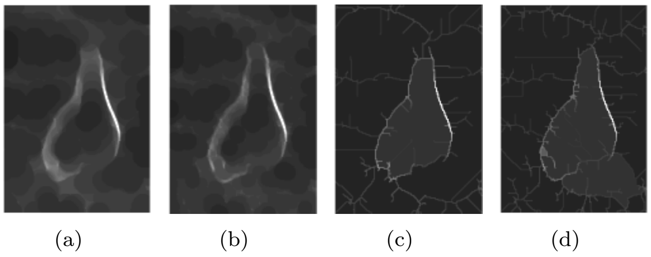


Figure 13. Effect of the viscous closings on the gradient image. (a) Mercury model. (b) Oil model. (c-d) Homotopic thinning of (a-b). Note that, only the oil model preserves the crest lines of low contrast.

Instead of summing the dilated sets, the distance function can be computed by translating and superposing the dilated sets:

$$\sum_{n=0}^{N-1} \delta_n(\chi) = \bigvee_{n=1}^{N} n.\delta_{N-n}(\chi) = \bigvee_{n=0}^{N} n.\delta_{N-n}(\chi), \tag{11}$$

where $n.\chi$ corresponds to the set X represented with the luminance n.

Suppose now that the set X is replaced by a cylinder of high N and let f denote this function. All level sets $X_h(f)$ are identical if $0 \leq h \leq N$ and empty for $h > N$. This example allows to rewrite the precedent formulation for functions as follow:

$$\bigvee_{h \geq 0} h.\delta_{N-h}(\chi_h(f)). \tag{12}$$

This expression as to be compared to the oil type viscous transformations:

$$T_N^v(f) = \bigvee_{h \geq 0} h.T_{N-h}(\chi_h(f)). \tag{13}$$

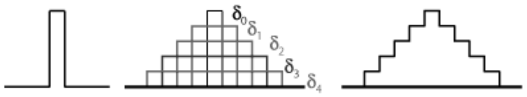


Figure 14. Principle of the viscous dilation (oil type of fluid). The original function level sets are dilated then stacked. The viscous dilation results from a supremum.

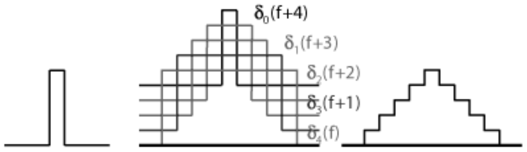


Figure 15. Viscous dilation computation (case of a mercury type of fluid). The original function is shifted then dilated. The original dilation results from an infimum.

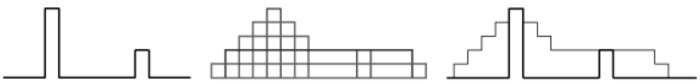


Figure 16. Effect of the viscous dilations on walls of different altitudes (case of a oil type of fluid).

The similitude is illustrated in Figure 14: dilations of decreasing activity are progressively stacked as it was the case for the oil model viscous transformations. As a consequence, distance functions may be interpreted as viscous transformations associated with dilations of decreasing activity. We will call this transformation *viscous dilation.*

By analogy with viscous closings two viscous dilations (inspired from the oil and the mercury models) may be defined:

$$\delta^v(f) = \bigvee_{h\geq 0} h.\delta_{N-h}(\chi_h(f)) \quad \text{and} \quad \widetilde{\delta^v}(f) = \bigwedge_{h\geq 0} \delta_{N-h}(f+h). \tag{14}$$

These transformations are illustrated on Figures 14, 15 and 16.

Of course, these transforms are equivalent to the sum $\sum_{n=0}^{N-1} \delta_n(f)$ for binary functions but the three transformations differ in the case of gray-scale functions.

As an illustration, the viscous transformations have been tested on gray-scale images: in Figure 17, the goal is the segmentation of the bird.

The bird is first roughly localized. Then, the gradient norm of the original function is computed. For reference, we present the segmentation obtained by computing the watershed transform directly on the original gradient image. A more robust segmentation can be obtained if a viscous dilation is applied on the gradient before computing the watershed transform. And

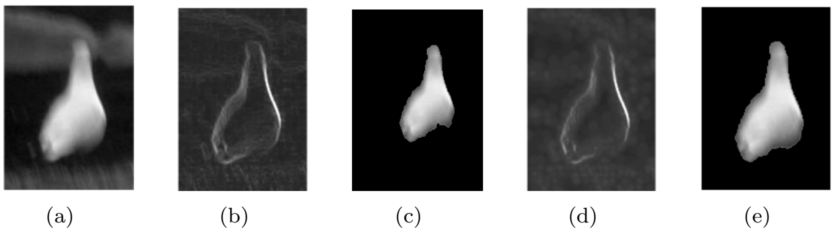


Figure 17. (a) Original image. (b) Gradient image. (c) Associated watershed transform. (d) Viscous dilation of the gradient image. (e)Associated viscous watershed transform (watershed computed after viscous closing of the relief).

a more regularized solution is obtained if, in addition to the viscous dilation, the standard watershed is replaced by the viscous watershed. We recall that the viscous watershed is a standard watershed computed after a viscous closing of the relief.

Another example is presented on Figure 18. The original image is first thinned, then a viscous dilation is computed and the derived image is thinned. So, one can observe how crest lines are connected. The result obtained via oil and mercury types of viscosity are compared. Lastly, watersheds are computed and compared.

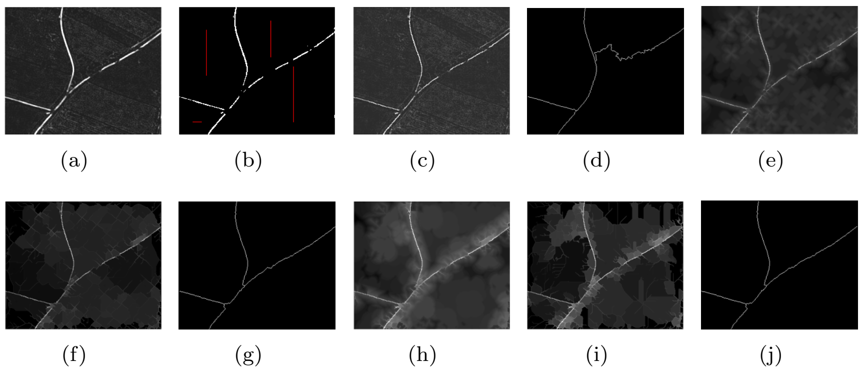


Figure 18. (a) Original image. (b) Markers. (c) Thinned original image. (d) Watershed segmentation. (e) Viscous dilation of the thinned original image (oil model). (f) Thinning of the result of the viscous dilation. (g) Watershed segmentation. (h-j) The same than (e-g) using the mercury model.

The viscous dilation allows to reconnect disconnected contours portions while viscous closings have a regularization effect. It is possible to combine the advantage of both dilations and closings by considering the family of the closing dilated sets. For the “oi” model, it expresses as:

$$\bigvee_{h \geq 0} \varphi_{r(h)} \delta_{r(h)}(h.\chi_h(f)).$$

5. Conclusion

Viscous transformations appear to be extensions to grey-tone functions of distance functions or opening function of binary sets (sum of dilations or openings of increasing size). As distance functions of binary sets are useful for connecting binary dots or separating particles, viscous dilations are useful for filling missing gaps in grey-tone contours. They may be used besides or in conjunction with viscous closings in order not only to fill in gaps but also to regularize the contours of gray-tone images. In a next paper we will further study the properties of these transformations.

References

[1] S. Beucher, *Segmentation d'images et morphologie mathématique*, Ecole des Mines de Paris, Fontainebleau, France, 1990.

[2] G. Matheron, *Eléments pour une Théorie des Milieux Poreux*, Masson, Paris, 1967.

[3] F. Meyer, *Skeletons and Perceptual Graphs*, Signal Processing **16** (1989), no. 4, 690–710.

[4] F. Meyer and C. Vachier, *Image Segmentation based on Viscous Flooding Simulation*, Mathematical Morphology and its Applications to Image and Signal Processing (2002), 69–77.

[5] ———, *On the Regularization of The Watershed Transform* (P. W. Hawkes, ed.), Academic Press, 2007, Advances in imaging and electron physics, pp. 195–251.

[6] J. Serra, *Viscous Lattices*, Journal of Mathematical Imaging and Vision **22** (2005), 269–282.

[7] P. Soille, *Morphological Image Analysis (Principles and Applications)*, Springer-Verlag, 1999.

[8] C. Vachier and F. Meyer, *The Viscous Watershed Transform*, Journal of Mathematical Imaging and Vision **22** (2005), 251–267.

A region-based interpolation method for mosaic images

Javier Vidal[1, 2], Jose Crespo[1] and Víctor Maojo[1]

[1] *GIB - LIA, Facultad de Informática, Universidad Politécnica de Madrid, Spain*
`{jvidal,jcrespo,vmaojo}@infomed.dia.fi.upm.es`

[2] *Departamento Ingeniería Informática y Ciencias de la Computación, Facultad de Ingeniería, Universidad de Concepcion, Chile*
`jvidal@udec.cl`

Abstract Image interpolation is an important operation in some image analysis and processing applications. This paper describes a region-based interpolation method for mosaic images. It is an extension of a previously presented interpolation technique based on morphological operations for binary images. Even though the principles used in this method are similar to those used in the technique for binary images, there are some substantial differences, pointed out in this paper, due the nature of images treated. Some experimental results are provided.

Keywords: mathematical morphology, image processing, image analysis, interpolation, mosaic, median set.

1. Introduction

Image interpolation is necessary in some image processing applications. Methods for image interpolation construct new images from a set of known ones, and they permit to increase the practical resolution of data. However, it should be stated that the interpolated images are obtained from the input images and depend completely on them.

Mathematical morphology [5, 16, 17, 20] techniques have been used to design several interpolation methods [1–4, 7–15, 18, 24] that deal with the shapes of the objects to be interpolated. In this paper, we are especially interested in morphological interpolation for mosaic images [10–12, 24].

The interpolation method for mosaics described in this paper is morphological as well. The principles used are similar to those used in an interpolation technique for binary images presented in [21–23]. These basic principles are two: an inclusion property and the utilization of the median set as interpolator [19]. Despite such similarities, the nature of the treated images make it necessary to treat some additional aspects, which are pointed out in the paper.

The proposed method is region-based. In this work, the input images are 2D mosaic images, i.e., 2D gray-level segmented images that are composed of piecewise-constant regions.

This paper is organized as follows. Section 2 presents a brief summary of the previously presented interpolation technique. Section 3 presents an extension of this technique in order to treat the so-called border images. Characteristics and technical details of this region-based interpolation method are described in Section 4. Some experimental results are provided in Section 5. Conclusions are commented in Section 6.

2. A brief summary of an interpolation technique for binary images

The technique reported in this paper is based on a previous work of ours applied to binary images [21–23]. It will be briefly described in this section.

2.1 Inclusion property

Our technique is based on an inclusion property that establishes a relationship which we think can greatly facilitate and improve the results of interpolation methods for binary images. It establishes the recursive interpolation of shapes with internal structures (pores and grains).

Formally, if A_i and B_i are two sets of input slice 1, such that $B_i \subset A_i$, and A_j and B_j are two sets of input slice 2, such that $B_j \subset A_j$, and we want to interpolate A_i with A_j, and B_i with B_j, then the following condition should be satisfied:

$$Inter(A_i \setminus B_i, A_j \setminus B_j) = Inter(A_i, A_j) \setminus Inter(B_i, B_j), \tag{1}$$

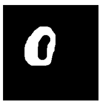
"*Inter*" corresponds to the interpolation computation. The arguments of *Inter* are the two images to be interpolated and its result is the interpolated one. Figure 1 illustrates inclusion property: Figure 1(b) is equal to Figure 1(e) minus Figure 1(h).

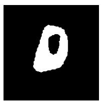
2.2 Binary image interpolation algorithm

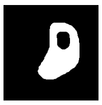
In the algorithm for interpolating binary images [21–23], we can distinguish three main sections: (1) separation of outer CCs from each slice, (2) matching of CCs (one from input slice 1 and another from input slice 2), and (3) interpolation of matched CCs. Note that in this technique CC refers to a connected component, and it can denote a grain or a hole.

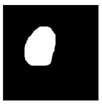
In the first step, the outer filled CCs of the input slices are identified and separated. The outer filled CCs are the filled CCs surrounded by the background pixels that touch the border of the image. Then, in the matching step, the method establishes correspondences between CCs from the different slices. Those CCs that match will be aligned in order to overlap them and, after that, interpolated using a median set computation.

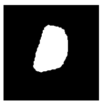
A detailed description of the algorithm pseudo-code for interpolating binary images can be found in [22].

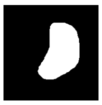

(a) Input set 1

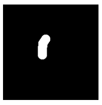

(b) Interpolated set

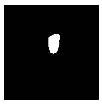

(c) Input set 2


(d) Filled grain 1


(e) Interpolated filled grain


(f) Filled grain 2


(g) Hole 1


(h) Interpolated hole

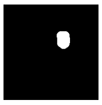

(i) Hole 2

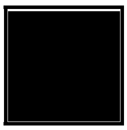
Figure 1. Inclusion property example. Part (d) is the filled grain of part (a). Similarly, part (f) is the filled grain of part (c). Parts (g) and (i) are the holes of, respectively, parts (a) and (c).

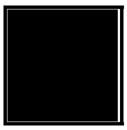
3. Treatment of border regions in mosaic images

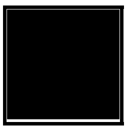
In the first step of the mosaic interpolation method, regions of connected pixels with the same gray-level are extracted from the pair of input slices and are converted to binary images. Some of the extracted regions touch the border of the images. These regions pose a distinctive problem that must be treated and satisfactorily solved. Next section describes how our technique deals with it. It should pointed out that this issue was not treated in [21–23]).

3.1 Classification of border images

A "binary border image" is an image that has a CC that touches its border. In formal terms, let I and X represent, respectively, a binary image and a connected component (clearly, $X \subseteq I$). If ∂I denotes the set of border


(a) ∂A_1


(b) ∂A_2


(c) ∂A_3

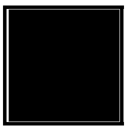

(d) ∂A_4

Figure 2. Border test images.

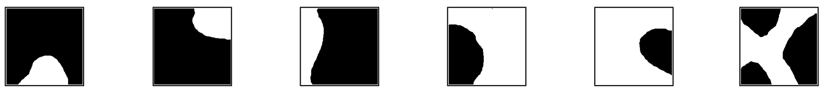
points of image I (i.e., if the image is $N \times M$, its border pixels are those located at row 0, row $(N-1)$, column 0 and column $(M-1)$), then I is a binary border image if $\partial I \cap X \neq \emptyset$. Figure 4 illustrates several examples of such images[1].

```
BORDERCLASS ($X$:Slice):integer {
  type = 0;
  IF numberOfCC($X^C$) > 1
    type = 8;
  ELSE
    FOR $i$ = 1 TO 4
      IF $| \partial A_i \cap X | = | \partial A_i |$
        type = type + 2;
      ELSEIF $| \partial A_i \cap X | \neq 0$
        type = type + 1;
  RETURN (type);
}
```

Figure 3. Classification function.

Border CCs are detected and processed by defining four test images, called ∂A_1, ∂A_2, ∂A_3 and ∂A_4. These images are illustrated in Figure 2, and they are composed of a line located at one of the four image borders (up, right, down or left).

The classification of border images is performed by a function described in Figure 3. It consists in intersecting the border and test images. In the classification function in Figure 3, the notation $| Y |$ denotes the cardinal of Y, i.e., the number of pixels of set Y. If X touches all the border ∂A_i, the "type" variable is incremented by 2; if X touches partially the border ∂A_i, "type" is incremented by 1. This is computed for each border. All types of border images are displayed in Figure 4. Note that *Type* 3 and *Type* 5 are not possible. Also, note that *Type* 0 (not illustrated in Figure 4) is a non-border image. We have arbitrarily defined as *Type* 8 the case in which a border CC touches more than one of the border test images ∂A_i and has an intersection with $\cup_i(\partial A_i \cap X)$, $i \in \{1, ..., 4\}$, that is not a connected set.



(a) Type 1 (b) Type 2 (c) Type 4 (d) Type 6 (e) Type 7 (f) Type 8

Figure 4. Border images classification.

[1]Images are surrounded by a gray-frame in order to distinguish the boundaries of the border CCs. Sets and CCs appear in white in this paper.

3.2 Interpolation of border images

Border CCs detected according to the procedure described in the last section should be interpolated. The interpolation of border CCs is described in the following. Several cases are distinguished. (Note: the region border type is used in case 4.) Let X_1 and X_2 be CCs in two different slices:

1. If X_1 and X_2 are border CCs, such that $X_1 \cap X_2 \neq \emptyset$ then interpolate with X_1 and X_2 in their original location (without alignment). Figure 5 illustrates an example of this situation. Input images are the first and the last one.

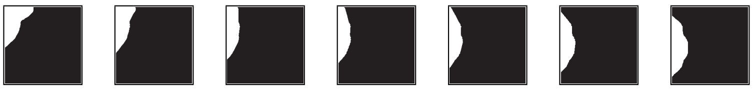


Figure 5. Interpolation between border images with non-empty intersection.

2. If just X_1 or X_2 is a border CC and if $X_1 \cap X_2 \neq \emptyset$ then interpolate with X_1 and X_2 in their original location. In Figure 6 is illustrated an example of this situation.

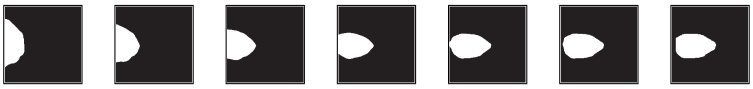


Figure 6. Interpolation between a border image and non-border image with non-empty intersection.

3. If either X_1 or X_2 is a border CC and $X_1 \cap X_2 = \emptyset$ but $\delta_{\lambda_1}(X_1) \cap X_2 \neq \emptyset$ or $X_1 \cap \delta_{\lambda_2}(X_2) \neq \emptyset$ (i.e., these CCs satisfies a proximity test (see later Section 4.2) and are considered a matched pair), then interpolate X_1 and X_2 normally. Figure 7 illustrates an example of this case.

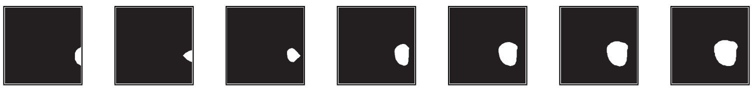


Figure 7. Interpolation between a border image and non border image with empty intersection.

4. In this case, either X_1 or X_2 is the empty set (i.e., it is non-existent). Let us suppose that X_2 is the empty set, so that X_1 vanishes from slide 1 to slide 2. This is dealt in [21–23] with the so-called "artificial" CCs,

so that an interpolation is performed between X_1 and an artificial point or line in slide 2. Figure 8 displays several cases of artificial CCs for different types of border CCs. (That is, in this case, the border region classification described in Section 3.1 is employed.) For example, in Figure 8, the Type 2 case uses as artificial CC a point (Figure 8(b)), and the Type 1 case employs a line at the bottom (Figure 8(a)).

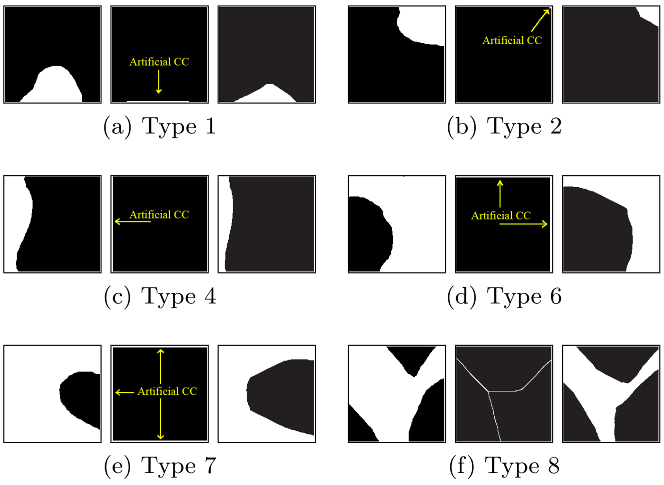


Figure 8. Artificial CCs to interpolate isolated border CCs: (a) $X \cap (\bigcup_{k=1}^{4} \partial A_k)$, (b) $X \cap ((\partial A_1 \cap \partial A_2) \cup (\partial A_2 \cap \partial A_3) \cup (\partial A_3 \cap \partial A_4) \cup (\partial A_4 \cap \partial A_1))$, (c), (d) and (e) $\bigcup_{k=1}^{4} X \cap \partial A_k$, if $X \cap \partial A_k = \partial A_k$, (f) $Thinning(A)$. In each case, the left image is X_1, the middle image is the artificial CC, and the right image is the interpolated result.

4. Interpolation of mosaic images

Mosaics are the gray-level images better suited to be treated by a region-based interpolation technique. Morphological interpolation for mosaic images is a recent subject. In the literature, the treatment of shapes in region-based interpolation problems have been frequently involved a conversion to binary images as strategy to treat shapes. In particular, binary morphological skeletons are used in [24] to interpolate shapes of regions. On the other hand, [11,12] describes a morphological mosaic interpolation based on the distance function calculation. This technique incorporates affine transformations in order to align matched regions (distance functions should be applied to non-empty intersected regions).

The general algorithm of ours region-based interpolation method is displayed in Figure 9. The algorithm is divided in 3 main parts: (1) detection and separation of regions in each slice, (2) matching and interpolation between regions, and (3) final adjustment. Note that these three steps are not exactly identical to the three steps of the binary image interpolation outlined in Section 2; the matching and interpolation stages have been integrated, and the final adjustment step is new (it was not necessary for the binary image case). The next sections describe these parts of the algorithm.

4.1 Region separation

In mosaics, a region corresponds to a set of connected pixels with the same gray-value. In the first step of our algorithm, step (1) in Figure 9, regions are extracted and stored as binary images. The gray-level value for each region is also stored, as well as the hierarchical level that corresponds to the region in a tree structure that is built. A simple mosaic and its regions structure is displayed in Figure 10. Figure 10(a) is the input mosaic, Figures 10(d) and 10(e) are two regions with gray-level value equal to 27, and Figures 10(f) to 10(h) are regions with gray-level values equal to 193, 130 and 93, respectively. Figures 10(b) and 10(c) illustrate the structure of the mosaic and the hierarchical region-based tree of the input image [6].

The first level is composed by all the regions which are directly or indirectly adjacent to the border of the image; the next level includes regions that belong to the internal structures of the regions in the first level in such a way that they are directly or indirectly adjacent to these regions, and so on.

This tree-based information was not necessary in binary images, because the only structure inside a CC that could exist would be a hole or grain inside (and not many regions).

4.2 Matching and interpolation

The objective of the matching step is to establish correspondences between (a) each region in slice 1 and (b) zero or more regions in slice 2. Such a number can be zero if no correspondence occurs. Regions to be matched must have an identical gray-level value as a prerequisite. The matching step corresponds to step (2) in the region-based interpolation algorithm shown in Figure 9.

The criteria used in the matching step are the proximity test and the minimal distances between their MSP[2] points. The proximity test consists in the computation of the *proximity zone.* The proximity zone of X is a dilation of X with a disk-shaped structuring element of radius λ_X. X's

[2]A filled CC, representing a filled region is reduced to a point using the Minimal Skeleton by Pruning [20], i.e., its skeleton is reduced by pruning until a final point is reached. For a filled CC definition see Section 2.2.

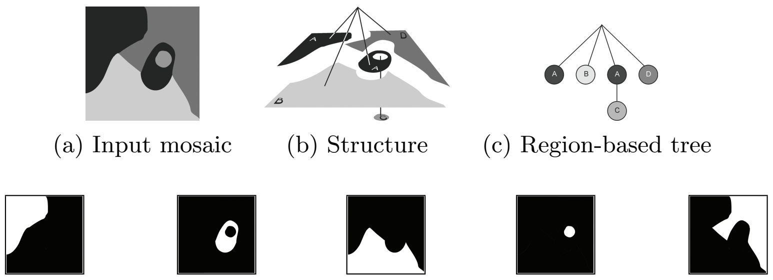
```
$INTERPOLATOR\ (S_1, S_2{:}Slice){:}Slice$ {
  // (1) Detection and separation of regions in each slice
  For each slice $S_i$
    Store region $j$ from slice $S_i$ into vector element $RS_i^j$
    Compute and store the hierarchical level of $RS_i^j$
  // (2) Matching and Interpolation (Regions are processed by hierarchical level)
  For each level $k$ of the hierarchical region tree
    For each region $i$ in slice $S_1$
      For each region $j$ in slice $S_2$
        If hierarchical levels of $RS_1^i$ and $RS_2^j$ are $k$
          // Only regions with the same gray-level are processed
          If graylevel of both $RS_1^i$ and $RS_2^j$ are equal
            If $RS_1^i$ and $RS_2^j$ pass the proximity test
                Establish matching between region $i$ and region $j$
    // If region i from slice 1 does not match any region in slice 2 or if region j from
    // slice 2 does not match any region in slice 1, the next level is searched
    For each region $i$ in slice $S_1$
      For each region $j$ in slice $S_2$
        If hierarchical level of $RS_1^i = k$ and hierarchical level of $RS_2^j = k+1$
        or hierarchical level of $RS_1^i = k+1$ and hierarchical level of $RS_2^j = k$
          If graylevel of both $RS_1^i$ and $RS_2^j$ are equal
            If $RS_1^i$ and $RS_2^j$ pass the proximity test
                Establish matching between region $i$ and region $j$
  // If there exists multiple matching, only regions with minimal distance
  // between them are kept
  For each region $i$ in slice $S_1$
    If multiple matching is detected for region $i$
      Choose to match region $k$ from slice $S_2$ with minimal MSP-distance from region $i$
  For each region $j$ in slice $S_2$
    If multiple matching is detected for region $j$
      Choose to match region $k$ from slice $S_1$ with minimal MSP-distance from region $j$
  // Interpolation between matched regions is performed computing median sets.
  For each region $i$ in slice $S_1$
    For each region $j$ in slice $S_2$
      If region $RS_1^i$ match with region $RS_2^j$
        Compute the interpolated region $S_p$ between $RS_1^i$ and $RS_2^j$ using median set
  // Isolated regions are interpolated with artificial regions (see Figure 8)
  For each region $i$ in slice $S_1$
    If region $RS_1^i$ is isolated
      Choose to match corresponding artificial region for region $RS_1^i$
      Calculate interpolated region $S_p$ between $RS_1^i$ and its artificial region
  For each region $j$ in slice $S_2$
    If region $RS_2^j$ is isolated
      Choose to match corresponding artificial region for region $RS_2^j$
      Calculate interpolated region $S_p$ between $RS_2^j$ and its artificial region
  // (3) Final adjustment
  Compute overlapped regions $\cup_{i=1,p-1} \cup_{j=i+1,p} (S_i \cap S_j)$ and empty regions $[\cup_{i=1,p} S_i]^C$
  Flood overlapped and empty regions with a watershed procedure
}
```

Figure 9. General region-based interpolation algorithm.

radius is computed as $\lambda_X = \min\{\alpha : X \subseteq \delta_\alpha(MSP_X)\}$, where MSP_X is the MSP of X.

If two regions X and Y from different slices match, i.e., $\delta_{\lambda_X}(X) \cap Y \neq \emptyset$ or $X \cap \delta_{\lambda_Y}(Y) \neq \emptyset$, the distance between MSP_X and MSP_Y is stored.

Multiple matching can happen, that is, some regions can match with



(a) Input mosaic (b) Structure (c) Region-based tree

(d) Region A (e) Region A (f) Region B (g) Region C (h) Region D

Figure 10. A mosaic divided in regions.

more than one region of the other slice. Sometimes, this kind of situations produces a merged region. This situation is solved in the following manner: let X be a region from slice 1 that match with n regions $Y_1, Y_2, ..., Y_n$ from slice 2. We must choose Y_k, such that,

$$Y_k = \min_{1 \leq i \leq n} \{Y_i, MSP\_distance(X, Y_i)\},$$

where $MSP\_distance$ represent the Euclidean distance between the MSP points of each input region. The rest of matched regions are rejected.

These criteria and procedures were already used in the interpolation method for binary images [21–23] but, in the case of mosaics, we consider also the hierarchical level of the regions. Particularly, two regions can match if the difference between their hierarchical level is at most one level. This condition is verified before the proximity test in the matching step. The hierarchical level is also used to compute the median sets, i.e., when all the regions at a certain hierarchical level are matched, the median sets between them are computed. After that, correspondences between regions at the next level are analyzed and so on.

Finally, isolated regions are treated. They correspond to regions that do not match and are therefore interpolated with an artificial region whose shape (a point or a line) depends on the position of the region (see examples in Figure 8).

4.3 Final adjustment

This stage is necessary for mosaic image interpolation but not for the binary image case (in fact, this stage was not present in our previous binary image interpolation technique [21–23]). In mosaics, after the previous steps we have a set of interpolated regions $\{R_i\}$ that pose two problems: (a) $\cup_i R_i$ does not necessarily cover the whole image support (i.e., there are empty spaces that do not belong to any R_i); and (b) in general, interpolated regions can overlap (i.e., pixels can belong to more than one interpolated region).

These problems are solved at step (3) of the general region-based interpolation algorithm shown in Figure 9. Pixels that belong to empty spaces or to overlapped spaces must be assigned to *one* region. This is performed by using a simple watershed procedure in this last part of the algorithm. Particularly, the image to be flooded is the complement of the image constituted by the interior of $\cup_i R_i$ minus the overlapping regions. Note also that appropriate gray-level values must be used to label the interpolated regions.

5. Experimental results

This section discusses some experimental results of our method. We have used synthetic images that permit to emphasize some relevant aspects of our method.

Figure 11 illustrates a simple case with a human-like mosaic. Input slices are the first and the last one. The body and the two background regions of the images are treated as border regions. Note that the body is Type 8, the big background is Type 6 and the small background is Type 2. The face, eyes and mouth regions are treated as CCs with inclusion relationships using the hierarchical region-based tree.

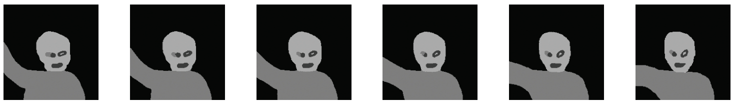


Figure 11. Example of mosaic interpolation.

The example in Figure 12 shows a situation where there exist an isolated region, e.g., the dark region situated at the bottom right of the image, which vanishes through from the first to the last input image. A border region (the white one at the upper left) that moves to the upper-left corner of the image is also displayed in this sequence.

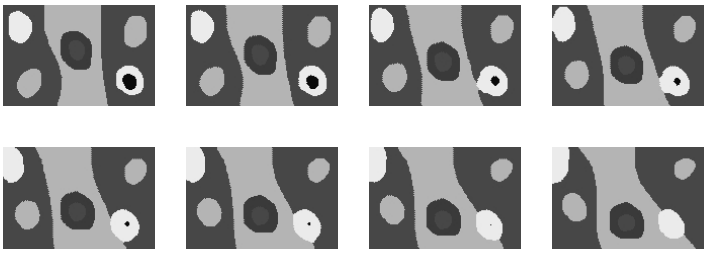


Figure 12. Example of mosaic interpolation.

6. Conclusions

In this paper, we have presented a region-based interpolation method for mosaic images. It is based on a previous interpolation technique of ours for binary images. As is shown in the paper, even though the basic principles in both techniques are similar, there exist some substantial differences between both cases. For example, the matching and interpolation operations are performed level by level of the hierarchical region-based tree that represents the region structure of the image. The overall interpolation is achieved when all levels have been processed.

Another important difference is that it is necessary a new final adjustment step for mosaic interpolation. After regions have been interpolated, there are generally pixels in the interpolated slice that do not belong to any region. In addition, there can be overlapping between interpolated regions. The last step of our interpolation algorithm for mosaics solve these problems.

Acknowledgments

This work has been supported in part by the University of Concepción (Chile), by the MECESUP programme of the Education Ministry of Chile, by the Artificial Intelligence Laboratory of School of Computer Science of "Universidad Politécnica de Madrid", and by the collaboration programme of Banco Santander-Central Hispano and Universidad Politécnica de Madrid for doctoral studies.

References

[1] S. Beucher, *Interpolations d'ensembles, de partitions et de fonctions*, Technical Report N-18/94/MM, Centre de Morphologie Mathématique, 1994.

[2] ———, *Interpolation of sets, of partitions and functions*, Mathematical morphology ant its applications to image and signal processing, 1998, pp. 119–126.

[3] A. G. Bors, L. Kechagias, and I. Pitas, *Binary morphological shape-based interpolation applied to 3D tooth reconstruction*, IEEE Transaction on Medical Imaging **21** (2002), no. 2.

[4] V. Chatzis and I. Pitas, *Interpolation of 3D binary images based on morphological skeletonizations*, IEEE Transaction on Medical Imaging **19** (2000), no. 7.

[5] E. R. Dougherty and R. A. Lotufo, *Hands-on morphological image processing*, SPIE Press, Bellingham, WA, 2003.

[6] L. Garrido, *Hierarchical region-based processing of images and video sequences: Application to filtering, segmentation and information retrieval*, Ph.D. Thesis, Department of Signal Theory and Communications - Universitat Politècnica de Catalunya, 2002.

[7] J.-F. Guo, Y.-L. Cai, and Y.-P. Wang, *Morphology-based interpolation for 3D medical image reconstruction*, Computarized Medical Imaging and Graphics **19** (1995), no. 3, 267–279.

[8] M. Iwanowski and J. Serra, *The morphological - affine object deformation*, International Symposium on Mathematical Morphology (ISMM), palo alto, CA, 2000, pp. 445.

[9] M. Iwanowski, *Application of mathematical morphology to image interpolation*, Ph.D. Thesis, School of Mines of Paris - Warsaw University of Technology, 2000.

[10] ______, *Generalized morphological mosaic interpolation and its application to computer-aided animations*, Computer analysis of images and patterns, 2001, pp. 494–501. Proceedings of 9th International Conference CAIP 2001, Sept. 5-7, 2001, Warsaw, Poland.

[11] ______, *Image morphing based on morphological interpolation combined with linear filtering*, International journal of WSCG, 2002, pp. 233–239. Proceedings of the 10th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, Pilzen, Czech Republic.

[12] ______, *Universal morphological interpolator* (I. S. P. Society, ed.), Vol. II, 2005. Proceedings of International Conference on Image Processing ICIPŠ05, Genova, Italy.

[13] T.-Y. Lee and W.-H. Wang, *Morphology-based three-dimensional interpolation*, IEEE Transactions on Medical Imaging **19** (2000), no. 7, 711–721.

[14] F. Meyer, *Interpolations*, Technical Report N-16/94/MM, Centre de Morphologie Mathématique, 1994.

[15] ______, *A morphological interpolation method for mosaic images*, Mathematical morphology and its applications to image and signal processing, 1996.

[16] J. Serra, *Mathematical morphology. Volume I*, London: Academic Press, 1982.

[17] ______ (ed.), *Mathematical morphology. Volume II: Theoretical advances*, London: Academic Press, 1988.

[18] ______, *Interpolations et distances of Hausdorff*, Technical Report N-15/94/MM, Centre de Morphologie Mathématique, 1994.

[19] P. Soille, *Spatial distributions from contour lines: an efficient methodology based on distance transformations*, Journal of Visual Communication and Image Representation **2** (1991), no. 2, 138–150.

[20] ______, *Morphological image analysis: Principles and applications*, 2nd ed., Springer-Verlag, 2003.

[21] J. Vidal, J. Crespo, and V. Maojo, *Inclusion relationships and homotopy issues in shape interpolation for binary images*, Lecture Notes in Computer Science, LNCS 3429, 2005, pp. 206–215.

[22] ______, *Recursive interpolation technique for binary images based on morphological median sets*, Computational Imaging and Vision Journal, 2005, pp. 53–62.

[23] ______, *A shape interpolation technique based on inclusion relationships and median sets*, Accepted in "Image and Vision Computing Journal" (2007).

[24] D. N. Vizireanu, S. Halunga, and O. Fratu, *A grayscale image interpolation method using new morphological skeleton*, Proceeding of 6th International Conference on Telecommunication in Modern Satellite, Cable and Broadcasting Services, 2003, pp. 519–521.

V

CONNECTIVITY

Adjacency stable connected operators and set levelings

JOSE CRESPO

GIB - LIA, Facultad de Informática, Universidad Politécnica de Madrid, Campus de Montegancedo, Boadilla del Monte (Madrid), Spain
jcrespo@fi.upm.es

Abstract This paper studies morphological connected operators. Particularly, it focuses on an adjacency constraint, as well as on the so-called set levelings. Two important findings are reported in this work. First, the relationships between the so-called adjacency stable operators and set levelings are investigated, and an equivalence is established. This is an important result about how these concepts have been chronologically introduced, and it permits to apply some properties to the related operator class. Second, the implications and limits of a property about expressing certain connected operators as a sequential composition of an opening and a closing (and vice-versa) based on markers are discussed. Then, a commutative property for attribute alternated filters is presented.

Keywords: image processing, image analysis, mathematical morphology, connected operator, adjacency stable operator, leveling.

1. Introduction

This paper investigates some aspects of morphological connected operators, which preserve well shapes and do not introduce discontinuities.

It happens that usual morphological connected operators (such as those composed by openings and closings) impose certain adjacency constraints between the input and the output. This was previously researched, and the so-called *adjacency stable connected* operator concept [6, 10, 11] was established in the set or binary framework.

In addition, the so-called *levelings* [20–23] were defined in the non-binary framework in such a way that constrain the output variations depending on the input variations. Those levelings within the set or binary framework are called set or binary levelings [22].

A question that naturally arises is whether there are relationships between adjacency stable connected operators and set levelings. As will be shown later, this is one of the results that will be established in this work. In scientific research, it is key being able to indicate how and when research concepts have been developed and introduced. Besides, in the case where such relationships exist, properties that are valid for one type of operators could perhaps apply to the other related operator class.

A second aspect researched in this paper is the possibility of expressing certain connected operators as the sequential composition of an opening and a closing (and vice-versa). Some clarifications about the implications, and limits, of a previously presented result are provided. Furthermore, a commutative property for attribute alternated filters will be presented.

Section 2 provides some general background. The adjacency stability and leveling concepts are recalled in, respectively, Sections 3 and 4. The relationships between adjacency stable connected operators and set levelings are established in Section 5. The expression of certain connected operators as sequential compositions of an opening and a closing is treated in Section 6. Finally, a conclusion section ends the paper.

2. Background

2.1 General definitions

Mathematical morphology deals with the application of set theory concepts to image processing and analysis, and it considers that images are composed of geometrical shapes with intensity or multi-band profiles [25]. Some general references are [1, 13, 15, 16, 19, 29–31, 35].

A basic set of notions on morphological filtering can be the following.

- Mathematical morphology focuses on increasing mappings defined on a complete lattice [31]. In a complete lattice there exists an ordering relation, and two basic operations called infimum and supremum (denoted by $\bigwedge$ and $\bigvee$, respectively).
- A transformation ψ is increasing if and only if it preserves ordering.
- A transformation ψ is idempotent if and only if $\psi\psi = \psi$.
- A transformation ψ is a morphological filter if and only if it is increasing and idempotent.
- An opening (denoted by γ) is an anti-extensive morphological filter. Since γ is anti-extensive, we can say that $\gamma \leq \text{id}$, where id symbolizes the identity operator that leaves the input unchanged.
- A closing (denoted by φ) is an extensive morphological filter. Since φ is extensive, we can say that $\varphi \geq \text{id}$.

In the theoretical expressions in this paper, we will be working on the lattice $\mathcal{P}(E)$, where E is a given set of points (the space) and $\mathcal{P}(E)$ denotes the set of all subsets of E (i.e., $\mathcal{P}(E) = \{A : A \subseteq E\}$). In other words, inputs and outputs are supposed to be sets or, equivalently, binary functions. In this lattice, the sup $\bigvee$ and the inf $\bigwedge$ operations are, respectively, the set union $\bigcup$ and set intersection $\bigcap$ operations, while the order relation is the set inclusion relation $\subseteq$. Even though we will work on the lattice $\mathcal{P}(E)$, results are extendable to gray-level functions by means of the so called flat operators [30, 35].

Two morphological operators ψ_1 and ψ_2 are dual if $\psi_1 = \complement\psi_2\complement$, where $\complement$ symbolizes the complement operator.

2.2 Some background on connectivity and connected operators

Some references to the topic of connectivity and connected filtering are: [2–5, 7, 9–12, 14, 17, 18, 20–24, 26–28, 32–38].

Connectivity is established in [31, (pp. 51–57)] by means of the *connected class* concept. A connected class $\mathcal{C}$ in $\mathcal{P}(E)$ is a subset of $\mathcal{P}(E)$ such that (a) $\emptyset \in \mathcal{C}$ and for all $x \in E$, $\{x\} \in \mathcal{C}$; and (b) for each family C_i in $\mathcal{C}$, $\bigwedge_i C_i \neq \emptyset$ implies $\bigvee_i C_i \in \mathcal{C}$. The subclass $\mathcal{C}_x$ that has all members of $\mathcal{C}$ that contain x (i.e., $\mathcal{C}_x = \{C \in \mathcal{C} : x \in C\}$) leads to the definition of an opening γ_x called *point opening* [31]. For all $x \in E$, $A \in P(E)$,

$$\gamma_x(A) = \bigvee\{C : C \in \mathcal{C}_x, C \leq A\}. \tag{1}$$

The dual operation of γ_x is the closing φ_x that is equal to $\complement\gamma_x\complement$. If a point x does not belong to a set A, i.e., it belongs to a pore of A, then we can obtain such a pore with $\gamma_x\complement$ (or, equivalently, with $\complement\varphi_x$).

In sets (or, equivalently, binary images), the *flat zone* of a point (or pixel) x is the grain or the pore (whichever is not empty) which x belongs to. I.e., the flat zone of x in a set A is equal to: $\mathrm{F}_x(A) = \gamma_x(A) \bigvee \gamma_x\complement(A)$. (Note that either the grain or the pore of a point x is empty.) In the non-binary case, the flat zones of a function are its piecewise-constant regions, i.e., the set of connected sets with the same function value.

An operator ψ is *connected* [28, 34] if, for all $A \in P(E)$, each flat zone (grain or pore) of A is included in a flat zone (grain or pore) of $\psi(A)$.

2.3 Connectivity requirement

A general requirement for the space connectivity is assumed in this work. Particularly, the space connectivity is supposed to be a *strong connectivity* (see [17, 24]). The usual four- and eight-connectivities in connected subsets of $\mathbb{Z}^2$ are cases of strong connectivities.

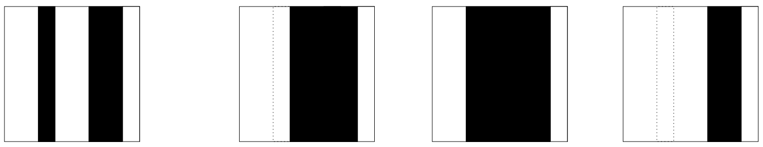
3. Adjacency stability

This section discusses and summarizes the adjacency stability concept for connected operators, which appeared first in [11], and was further studied in [6, 10]. A closely related concept was later discussed in [17].

Let us define the concept of *adjacency between flat zones*, which formalizes the intuitive notion of contiguity.

Definition 1. Two (disjoint) flat zones $\mathrm{F}_x(A)$ and $\mathrm{F}_y(A)$ in a space E (endowed with γ_x, $x \in E$) are said to be **adjacent** if $\mathrm{F}_x(A) \bigvee \mathrm{F}_y(A)$ is a connected set, i.e., if $\mathrm{F}_x(A) \bigvee \mathrm{F}_y(A) = \gamma_x(\mathrm{F}_x(A) \bigvee \mathrm{F}_y(A))$. □

(Note that $\mathrm{F}_x(A) = \gamma_x(A) \bigvee \gamma_x\complement(A)$.) The relationship of Definition 1 is symmetric (and not reflexive). Definition 1 can be extended to sets: two sets A and B are adjacent if some flat zone of A is adjacent to some flat zone of B.



(a) Input set (in dark) (b) AU Output (c) AS Output (d) AS Output

Figure 1. Adjacency stability example. Part (a) shows an input set A, and parts (b), (c) and (d) display three possible outputs of connected operators. Part (b) would be adjacency unstable: note that a pore of the input set A is in a grain in (b) but is not surrounded by grains of A. This situation does not happen in cases (c) and (d). (Note: "AU" refers to adjacency unstable, and "AS" denotes adjacency stable.) (Note: the space is endowed with a usual four- or eight-connectivity.)

The *adjacent flat zones* of a point x in an input set A, symbolized by $\mathrm{D}_x(A)$, are the pores (if $x \in A$) or the grains (if $x \notin A$) that are adjacent to $\mathrm{F}_x(A)$, i.e.,

$$\mathrm{D}_x(A) = \bigvee_y \{\mathrm{F}_y(A) : y \in E, \mathrm{F}_y(A) \bigvee \mathrm{F}_x(A) = \gamma_x(\mathrm{F}_y(A) \bigvee \mathrm{F}_x(A)\}.$$

The important concept of *adjacency stability* [6, 10, 11] is established next. This requirement concerns how adjacent grains and pores are treated by an operation.

Definition 2. Let E be a space endowed with γ_x, $x \in E$. An operator $\psi : \mathcal{P}(E) \Rightarrow \mathcal{P}(E)$ is **adjacency stable** if, for all $x \in E$:

$$\gamma_x(\mathrm{id} \bigvee \psi) = \gamma_x \bigvee \gamma_x \psi. \tag{2}$$

□

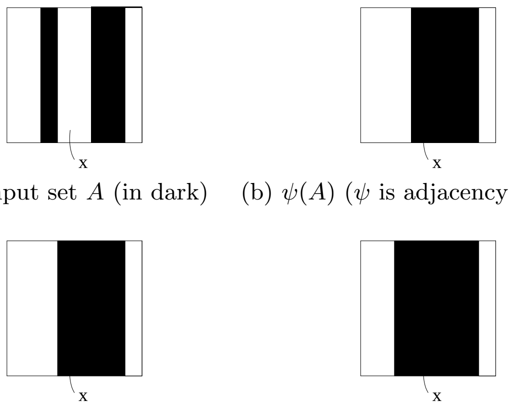
Note that γ_x commutes under the inf ($\gamma_x(\bigwedge_i \psi_i) = \bigwedge_i \gamma_x \psi_i$) but not in general under the sup.

The adjacency stability equation 2 treats grains and pores symmetrically. The reason is that what matters is the switch from grain to pore and vice-versa. We can state as well that the dual of an adjacency stable operator is adjacency stable.

The grain-pore relationship is illustrated in Figure 1. The consequences of adjacency stability on the relationships between the grains of an input set A and the output $\psi(A)$ are the following: the grains of $\psi(A)$ are a union of (a) grains of A, and (b) pores of A surrounded by grains in (a). For the particular case in Figure 1(b), Figure 2 shows that the adjacency stability equation does not hold for the point marked as x (this point is not the only one). The adjacency stability equation must hold for all $A \in \mathcal{P}(E)$ and for all $x \in E$.

Lemma 1 is useful to relate the input and the output.

Lemma 1. *Let E be a space endowed with γ_x, $x \in E$. A connected operator $\psi : \mathcal{P}(E) \Rightarrow \mathcal{P}(E)$ is adjacency stable if and only if, for all $A \in \mathcal{P}(E)$, $\psi(A)$ and $A \setminus \psi(A)$ are not connected to each other (i.e., are not adjacent).*



(a) Input set A (in dark) (b) $\psi(A)$ (ψ is adjacency unstable)

(c) $\gamma_x(A) \bigvee \gamma_x\psi(A)$ (d) $\gamma_x(\mathrm{id} \bigvee \gamma_x\psi)(A)$

Figure 2. Adjacency stability equation. Parts (a) and (b) display, respectively, an input set A and the output $\psi(A)$, where ψ is an adjacency unstable connected operator. The adjacency stability equation does not hold: for the point x signaled (among others), $\gamma_x(A) \bigvee \gamma_x\psi(A)$ (part (c)) is not equal to $\gamma_x(\mathrm{id} \bigvee \gamma_x\psi)(A)$ (part (d)).

Property 1. *Extensive and anti-extensive mappings are adjacency stable.*

Property 2. *The class of adjacency stable connected operators is closed under the sup, the inf and the sequential composition operations.*

If ψ is an adjacency stable connected operator, then, for all $A \in \mathcal{P}(E)$: (a) $x \notin A$, $x \in \psi(A) \Rightarrow \mathrm{D}_x(A) \leq \psi(A)$; (b) $x \in A$, $x \notin \psi(A) \Rightarrow \mathrm{D}_x(A) \leq \complement\psi(A)$.

4. Levelings and set levelings

A *leveling* [20–23] is defined next.

Definition 3. An image g is a leveling of an input image f if and only if:

$$\forall\ (p,q) \text{ neighboring pixels} : g_p > g_q \ \Rightarrow\ f_p \geq g_p \text{ and } g_q \geq f_q. \tag{3}$$

□

The previous definition of leveling is that in [20, Definition 4 (p. 193)] [23, Definition 2.2 (p. 4)]. A more general and complex definition is introduced in [22, Definition 10 (p. 62)], but a leveling as established by Expression 3 is also a leveling as defined in [22, Definition 10 (p. 62)]. Moreover, we focus on set operators.

Set levelings are those defined in the set or binary framework. Expression 3 will be particularized for the set or binary case in Section 5.2.

5. Relationships between adjacency stable connected operators and set levelings

Both adjacency stable connected operators and levelings impose certain constraints on the input and output variations, which will analyzed next. Then, the relationships between these types of operators will be discussed.

5.1 Adjacency stable connected operators: input/output variations restriction

First, we will formalize the neighborhood relationship between two pixels using γ_x. As in Section 4 (Definition 3), we will use the symbols p and q to refer to a pair of pixels that are neighbors. Two pixels p and q are *neighbors* to each other if and only if they satisfy

$$\gamma_p(\{p,q\}) = \gamma_q(\{p,q\}) = \{p,q\}. \tag{4}$$

The next proposition, which states the restrictions imposed on the values of neighboring pixels between an input I and the output I' of an adjacency stable connected operator, follows directly from Definition 2.

Property 3. *Let A be an input set. Let $\psi : \mathcal{P}(E) \Rightarrow \mathcal{P}(E)$ be an adjacency stable connected operator. If p and q are neighbors to each other (i.e., p and q satisfy equation 4), or if they belong to adjacent flat zones, the adjacency stability of ψ implies the following restrictions between the input A and output $\psi(A)$ sets:*

$$\begin{array}{lll} \text{(a)} \quad \gamma_p(A) = \gamma_q(A) & \Rightarrow & \gamma_p\psi(A) = \gamma_q\psi(A), \\ \text{(b)} \quad p \in A, q \notin A & \Rightarrow & \left\{ \begin{array}{c} p \in \psi(A), q \notin \psi(A) \\ \text{or} \\ \gamma_p\psi(A) = \gamma_q\psi(A). \end{array} \right. \end{array} \tag{5}$$

There is a case symmetric to (b) above (interchanging p and q in (b)) not shown.

The cases not covered in Expression 5 are: (i) $p \in A, q \notin A$ and $p \notin \psi(A), q \in \psi(A)$); and (ii) $p \notin A, q \in A$ and $p \in \psi(A), q \notin \psi(A)$. (Cases (i) and (ii) are symmetric to each other, interchanging p and q.) Clearly, these cases do not satisfy Equation 2 of Definition 2. Let us prove that for one case. We can prove that the case $p \notin A, q \in A$ and $p \in \psi(A), q \notin \psi(A)$) does not satisfy Equation 2 . We have that:

- $\gamma_p(\mathrm{id} \bigvee \psi)(A) = \gamma_p(A \bigvee \psi(A)) = \gamma_p\psi(A) \bigvee \gamma_q(A)$,
- $\gamma_p(A) \bigvee \gamma_p\psi(A) = \gamma_p\psi(A)$,

and $\gamma_p\psi(A) \bigvee \gamma_q(A) > \gamma_p\psi(A)$, since $q \in A$, $\gamma_q(A) \neq \emptyset$, and $\gamma_q(A) \nleq \gamma_p\psi(A)$.

We can straightforwardly extend Property 3 to binary functions (which can be considered as equivalent to sets). Let us assume that the binary values are 0 and 1. Let I and I' be, respectively, an input image and the output of an adjacency stable connected operator ψ (i.e., $I' = \psi(I)$).

Binary functions case: If p and q are neighbors to each other (or if they belong to adjacent flat zones), then:

$$\begin{array}{lll} \text{(a)} \quad I_p = I_q & \Rightarrow & I'_p = I'_q, \\ \text{(b)} \quad I_p = 1, I_q = 0 & \Rightarrow & \left\{ \begin{array}{c} I'_p = 1, I'_q = 0 \\ \text{or} \\ I'_p = I'_q. \end{array} \right. \end{array} \tag{6}$$

Note: the case symmetric to (b) (interchanging p and q in (b)) is not shown.

The cases ruled out are: (i) $I_p = 1, I_q = 0$, and $I'_p = 0, I'_q = 1$; and (ii) $I_p = 0, I_q = 1$, and $I'_p = 1, I'_q = 0$. (Cases (i) and (ii) are symmetric to each other, interchanging p and q.)

5.2 Set levelings: input/output variation restriction

Definition 3 can of course be applied to binary images (or, equivalently, sets). If we particularize for binary images then we have the following, concerning Expression 3. Similarly as in Section 5.1, I and I' denote the input and output images (i.e., I and I' have been substituted for f and g, respectively, in Expression 3). An inequality such as $I'_p > I'_q$ can only occur when there is a discontinuity where I'_p and I'_q are 1 and 0, respectively. Then, Expression 3 can only be:

$$1 > 0 \ \Rightarrow \ 1 \geq 1 \text{ and } 0 \geq 0, \tag{7}$$

i.e., I_p has to be 1, and I_q must be 0.

Therefore, if $I'_p = 1$ and $I'_q = 0$, the case where $I_p = 0$ and $I_q = 1$ is excluded.

5.3 Discussion

We can now precisely state the relationships between stable connected operators and set levelings: (a) Both adjacent stable connected operators and set levelings impose restrictions on the input/output variations. (b) The imposed restrictions are identical in both cases: if $I'_p = 1$ and $I'_q = 0$, then I_p and I_q must be 1 and 0, respectively.

Thus, **the set leveling concept and the adjacency stable connected operator concept are equivalent**. The adjacency stable connected operator notion [6, 10, 11] is prior in time to levelings [20, 21, 23] [22].

A final question can arise: can the adjacency stable connected operator concept be applied to a gray-level framework? The answer is affirmative, we can directly extend such input and output variation restrictions to flat gray-level connected operators that commute with thresholding, where the operator is level-by-level connected and where Equation 2 must hold for all sections of the input and output gray-level functions. Let I and I' be, respectively, an input image and the output of a flat gray-level adjacency stable connected operator ψ (i.e., $I' = \psi(I)$).

Gray-level (non-binary) functions case: If p and q are neighbors to each other (or if they belong to adjacent flat zones), then:

$$\begin{array}{lll} \text{(a)} \quad I_p = I_q & \Rightarrow & I'_p = I'_q, \\ \text{(b)} \quad I_p > I_q & \Rightarrow & \left\{ \begin{array}{c} I'_p > I'_q \\ \text{or} \\ I'_p = I'_q. \end{array} \right. \end{array} \tag{8}$$

Note: the case symmetric to (b) is not shown.

The case ruled out is: $I_p < I_q$, and $I'_p > I'_q$ (as well as the symmetric one: $I_p > I_q$, and $I'_p < I'_q$). This case is also excluded by Expression 3 of levelings. Moreover, disregarding trivial cases (such as those where, processing level by level, $\psi(\emptyset) = E$ or $\psi(E) = \emptyset$), in flat morphological (increasing) connected operators the variation range of the output is equal or smaller than that of the input. Otherwise, the flat zone inclusion relationship of connected operators would not necessarily hold between sections at same level of the input and output images. For example, in the case where $I_p > I_q$ and $I'_p > I'_q$, the next gradation would exist : $I_p \geq I'_p > I'_q \geq I_q$.

Now that the equivalence of adjacency stable connected operators and set levelings has been established, researchers know that the adjacency stable connected operator properties also hold for (flat) levelings. Let us point out that the properties included in Section 3 (Lemma 1 and Properties 1 and 2), altogether with the adjacency stability equation 2, are useful for manipulating expressions composed of connected operators, especially when they are also connected-component local (i.e., they treat grains or pores independently from the rest) [6, 10, 11]. In fact, they will be employed in a proof in the following section.

6. On commutative properties of open-close and close-open filters for connected operators

In this section we will first comment on an interesting commutative property that has been presented elsewhere, and on some of its implications and limits. Afterwards, we will also present another commutative property that is satisfied by certain alternated connected filters, particularly by attribute alternated filters.

6.1 Marker-based operators

Let us define the "$\|$" relationship presented in [33, Definition 7.3]:

Definition 4. Let A and B be, respectively, a grain (a connected set) and a set. We say that $A \parallel B$ if A and B have a non-empty intersection or are adjacent. □

Now, we are going to define two operators, $\overline{\gamma}$ and $\underline{\varphi}$, based on makers that are presented in [33, p. 176]. Besides one normal input set, those operators use a second one, which is a marker set.

Definition 5. Let A and M be two sets. The connected operator $\overline{\gamma}$ of a set A based on marker M, symbolized by $\overline{\gamma}(A, M)$, is defined as:

$$\overline{\gamma}(A, M) = \bigcup\{\gamma_x(A) : \gamma_x(A) \parallel M\}. \tag{9}$$

□

Note that Definition 5 does not define $\overline{\gamma}(A)$ but $\overline{\gamma}(A, M)$.

Definition 6. Let A and M be two sets. The connected operator $\underline{\varphi}$ of a set A based on marker N, symbolized by $\underline{\varphi}(A, N)$, is defined as:

$$\complement[\underline{\varphi}(A, \complement N)] = \bigcup\{\gamma_x(\complement A), x \in E : \gamma_x(\complement A) \parallel N.\} \tag{10}$$

□

In [33], it is established that, under some connectivity considerations, there exists a commutative property for $\overline{\gamma}$ and $\underline{\varphi}$ [33, Theorem 7.3]:

$$\overline{\gamma}(\underline{\varphi}(A, \complement N), M) = \underline{\varphi}(\overline{\gamma}(A, M), \complement N). \tag{11}$$

It is indicated in [33] that $\overline{\gamma}(\underline{\varphi}(A, \complement N), M)$ (or $\underline{\varphi}(\overline{\gamma}(A, M), \complement N)$) is a leveling and that is a strong filter. Moreover, in [22,23], it is mentioned that levelings are strong filters (see [23, Section 3.4 (p. 9)] and [22, Section 4.4.1 (p. 67)]), and the discussion refers to a commutative expression similar to the aforementioned one. This should be clarified, because it seems there could be some confusion about levelings (Definition 3), particularly about whether all levelings can be formulated as sequential compositions of an opening and a closing, and vice-versa.

There is an important remark, which concerns the main cause of possible misconceptions, to be made about Equation 11: the computation of the markers is *not* considered. Thus, **saying that an operation can be expressed as in Equation 11 does not necessarily imply that such an operation can be expressed as a sequential composition of an opening and a closing, and vice-versa, *when* the marker computation is considered**. For example, in $\overline{\gamma}(\underline{\varphi}(A, \complement N), M)$, the marker M of $\overline{\gamma}$ is not affected by the result of the previously applied $\underline{\varphi}$; that is, the M marker can be considered as an input. This is not how we usually think about openings and closings. In fact, when the marker computation is adequately considered, not all levelings (Definition 3) can be expressed as a sequential composition of an opening and a closing, and vice-versa; and it would not be the case that all levelings are strong filters. (See [9] [8] for further discussion about the strong property of connected alternated filters.)

Next section will present a commutative property for certain connected filters that does not raise that objection but where the underlying marker is, as usual, not considered fixed.

6.2 A commutative property

We present a commutative property for a certain type of connected alternated filters, particularly for alternated attribute filters, which are strong filters composed of an *attribute opening* [31, 34] [38] [35] and an *attribute closing.* Area openings and closings are examples of attribute openings and closings, respectively.

Let $\tilde{\gamma}$ and $\tilde{\varphi}$ denote, respectively an attribute opening and closing. In the following, unlike in Equation 11, when we write $\tilde{\varphi}\tilde{\gamma}$ it is clear that the criterion (and associated marker) of $\tilde{\varphi}$ is applied to the output computed by the previous $\tilde{\gamma}$. We have the following property:

Property 4. *An attribute alternated filter $\tilde{\varphi}\tilde{\gamma}$ can be expressed as a commutative sequential composition of an opening and a closing as follows:*

$$\tilde{\varphi}\tilde{\gamma} = \tilde{\gamma}\,(\mathrm{id} \bigvee \tilde{\varphi}\tilde{\gamma}) = (\mathrm{id} \bigvee \tilde{\varphi}\tilde{\gamma})\,\tilde{\gamma}. \tag{12}$$

Proof. There are two equalities to consider.

(a) Let A be a set. Since $\tilde{\gamma}$ is connected-component local we have $\tilde{\gamma} = \bigvee_x \gamma_x\tilde{\gamma} = \bigvee_x \tilde{\gamma}\gamma_x$. Thus, $\tilde{\gamma}(\mathrm{id} \bigvee \tilde{\varphi}\tilde{\gamma})(A) = \bigvee_x \gamma_x\tilde{\gamma}(\mathrm{id} \bigvee \tilde{\varphi}\tilde{\gamma})(A) = \bigvee_x \tilde{\gamma}\gamma_x(\mathrm{id} \bigvee \tilde{\varphi}\tilde{\gamma})(A)$. From Property 1 and Property 2, $\tilde{\varphi}\tilde{\gamma}$ is adjacency stable, and, from Lemma 1, $\tilde{\varphi}\tilde{\gamma}(A)$ and $A \setminus \tilde{\varphi}\tilde{\gamma}(A)$ are not adjacent [6, 10]. Then,

$$\tilde{\gamma}\gamma_x(\mathrm{id} \bigvee \tilde{\varphi}\tilde{\gamma})(A) = \begin{cases} \tilde{\gamma}\gamma_x(A) = \emptyset, & x \in A \setminus \tilde{\varphi}\tilde{\gamma}(A), \\ \tilde{\gamma}\gamma_x\tilde{\varphi}\tilde{\gamma}(A), & x \in \tilde{\varphi}\tilde{\gamma}(A). \end{cases} \tag{13}$$

Thus, $\bigvee_x \tilde{\gamma}\gamma_x\tilde{\varphi}\tilde{\gamma} = \bigvee_x \gamma_x\tilde{\gamma}\tilde{\varphi}\tilde{\gamma} = \tilde{\gamma}\tilde{\varphi}\tilde{\gamma}$. Finally, $\tilde{\gamma}\tilde{\varphi}\tilde{\gamma} = \tilde{\varphi}\tilde{\gamma}$ (since $\tilde{\varphi}\tilde{\gamma} \leq \tilde{\gamma}\tilde{\varphi}$ and $\tilde{\gamma}\tilde{\varphi}\tilde{\gamma} = \tilde{\varphi}\tilde{\gamma}$ [28, 34]).

(b) $(\mathrm{id} \bigvee \tilde{\varphi}\tilde{\gamma})\,\tilde{\gamma} = \tilde{\gamma} \bigvee \tilde{\varphi}\tilde{\gamma}\tilde{\gamma} = \tilde{\gamma} \bigvee \tilde{\varphi}\tilde{\gamma} = \tilde{\varphi}\tilde{\gamma}$.

□

Notes: (a) $(\mathrm{id} \bigvee \tilde{\varphi}\tilde{\gamma})$ is a closing (and different from $\tilde{\varphi}$). (b) Property 4 is different from [17, Proposition 10.2]. (c) This proof also provides an example of using adjacent stable connected operators properties to manipulate expressions. (d) Concerning filter expressions and decompositions, see also [7].

7. Conclusion

This paper has focused on connected morphological operators. First, the relationship between adjacency stable connected operators and set levelings has been investigated, and a close relationship, an equivalence, has been identified. This is important to establish how and when this concept has been introduced, and to clarify its origin. In addition, properties satisfied by one class of operators can be applied to the other equivalent one.

Second, this work has analyzed a commutative property previously presented, as well as some of its implications and limits. The paper has presented also a commutative property for attribute alternated filters, in which, as is usually the case and unlike in the other commutative property analyzed, the underlying marker computation is taken into account.

Acknowledgements

The author wants to thank the comments of Dr. Ronse about strong connectivities, as well as the remarks of an anonymous reviewer.

References

[1] G. J. F. Banon, *Formal introduction to digital image processing*, INPE, São José dos Campos, 2000. Available from: <`http://urlib.net/dpi.inpe.br/banon/1999/06.21.09.31`>. Access in: 2007-02-17.

[2] U. Braga-Neto, *Multiscale connected operators*, Journal of Mathematical Imaging and Vision **22** (2005), no. 2 - 3, 199–216.

[3] U. Braga-Neto and J. Goutsias, *A multiscale approach to connectivity*, Computer Vision and Image Understanding **89** (2003), no. 1, 70–107.

[4] ———, *A theoretical tour of connectivity in image processing and analysis*, Journal of Mathematical Imaging and Vision **19** (2003), no. 1, 5–31.

[5] E. J. Breen and R. Jones, *Attribute openings, thinnings, and granulometries*, Computer Vision and Image Understanding **64** (1996), no. 3, 377–389.

[6] J. Crespo, *Morphological connected filters and intra-region smoothing for image segmentation*, Ph.D. Thesis, 1993.

[7] J. Crespo and V. Maojo, *New results on the theory of morphological filters by reconstruction*, Pattern Recognition **31** (1998), no. 4, 419–429.

[8] ———, *The strong property of morphological connected alternated filters*, submitted for publication (2007).

[9] J. Crespo, V. Maojo, J. A. Sanandrés, H. Billhardt, and A. Muñoz, *On the strong property of connected open-close and close-open filters*, Discrete geometry for computer imagery, 2002, pp. 165–174.

[10] J. Crespo and R. W. Schafer, *Locality and adjacency stability constraints for morphological connected operators*, Journal of Mathematical Imaging and Vision **7** (1997), no. 1, 85–102.

[11] J. Crespo, J. Serra, and R. W. Schafer, *Image segmentation using connected filters*, Workshop on mathematical morphology, Barcelona, 1993, pp. 52–57.

[12] ———, *Theoretical aspects of morphological filters by reconstruction*, Signal Processing **47** (1995), no. 2, 201–225.

[13] E. R. Dougherty and R. A. Lotufo, *Hands-on morphological image processing*, SPIE Press, Bellingham, 2003.

[14] L. Garrido, P. Salembier, and D. Garcia, *Extensive operators in partition analysis for image sequence analysis*, Signal Processing **66** (1998), no. 2, 157–180.

[15] C. Giardina and E. Dougherty, *Morphological methods in image and signal processing*, Prentice-Hall, Englewood Clliffs, 1988.

[16] H. Heijmans, *Morphological image operators (advances in electronics and electron physics; Series Editor: P. Hawkes)*, Academic Press, Boston, 1994.

[17] ______, *Connected morphological operators for binary images*, Computer Vision and Image Understanding **73** (1999), 99–120.

[18] R. Jones, *Connected filtering and segmentation using component trees*, Computer Vision and Image Understanding **75** (1999), no. 3, 215–228.

[19] G. Matheron, *Random sets and integral geometry*, Wiley, New York, 1975.

[20] F. Meyer, *From connected operators to levelings*, Mathematical morphology and its applications to image and signal processing, 1998, pp. 191–198.

[21] ______, *The levelings*, Mathematical morphology and its applications to image and signal processing, 1998, pp. 199–206.

[22] ______, *Levelings, image simplification filters for segmentation*, Journal of Mathematical Imaging and Vision **20** (2004), no. 1-2, 59–72.

[23] F. Meyer and P. Maragos, *Nonlinear scale-space representation with morphological levelings*, Journal of Visual Communication and Image Representation **11** (2000), no. 3, 245–265.

[24] C. Ronse, *Set-theoretical algebraic approaches to connectivity in continuous or digital spaces*, Journal of Mathematical Imaging and Vision **8** (1998), no. 1, 41–58.

[25] ______, *Guest editorial*, Journal of Mathematical Imaging and Vision **22** (2005), no. 2 - 3, 103–105.

[26] C. Ronse and J. Serra, *Geodesy and connectivity in lattices*, Fundamenta Informaticae **46** (2001), no. 4, 349–395.

[27] P. Salembier and L. Garrido, *Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval*, IEEE Transactions on Image Processing **9** (2000), no. 4, 561–575.

[28] P. Salembier and J. Serra, *Flat zones filtering, connected operators, and filters by reconstruction*, IEEE Transactions on Image Processing **4** (1995), no. 8, 1153–1160.

[29] M. Schmitt and J. Mattioli, *Morphologie mathématique*, Masson, Paris, 1993.

[30] J. Serra, *Mathematical morphology. Volume I*, Academic Press, London, 1982.

[31] ______ (ed.), *Mathematical morphology. Volume II: Theoretical Advances*, Academic Press, London, 1988.

[32] ______, *Connectivity in complete lattices*, Journal of Mathematical Imaging and Vision **9** (1998), no. 3, 231–251.

[33] ______, *Connections for sets and functions*, Fundamenta Informaticae **41** (2000), no. 1-2, 147–186.

[34] J. Serra and P. Salembier, *Connected operators and pyramids*, Proceedings of SPIE, non-linear algebra and morphological image processing, san diego, 1993, pp. 65–76.

[35] P. Soille, *Morphological image analysis*: *Principles and applications*, 2nd., Springer-Verlag, Berlin-Heidelberg-New York, 2003.

[36] I. R. Terol and D. Vargas, *A study of openings and closings with reconstruction criteria*, Proc. of international symposium on mathematical morphology (ISMM) 2002, 2002.

[37] D. Vargas-Vázquez, J. Crespo, and V. Maojo, *Morphological image reconstruction with criterion from labelled markers*, Discrete geometry for computer imagery, 2003, pp. 475–484.

[38] L. Vincent, *Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms*, IEEE Transactions on Image Processing **2** (1993), 176–201.

Perceptual filtering with connected operators and image inpainting

Mariella Dimiccoli and Philippe Salembier

Technical University of Catalonia (UPC), Barcelona, SPAIN
{mariella,philippe}@gps.tsc.upc.edu

Abstract This paper focuses on a class of morphological filtering tools called connected operators. These operators act by merging flat zones that do not fulfill a given simplification criterion. This filtering approach offers the advantage of simplifying the image, because some flat zones are removed, as well as preserving contour information, because the flat zones that are not removed are perfectly preserved. However, for some applications, connected operators present a drawback in the way they restore the areas where flat zones have been merged. These areas may be perceptible in the filtered image appearing as single flat zones inside smoothed regions or across edges. To overcome such drawback, this paper proposes a solution based on the use of image inpainting.

Keywords: connected operator, image inpainting.

1. Introduction

In the context of image segmentation or region-based image analysis, the purpose of a filter is often to remove some image details that do not fulfill a certain simplification criterion. Many classical image filtering strategies are based on the use of a specific signal $h(x)$, such as an impulse response or a structuring element, which modifies the pixel values in a local window. However, these filters introduce distortions in the output because the signal $h(x)$ is not related at all with the input signal. A completely different approach is taken by morphological connected operators [17]. These operators act directly on the partition of flat zones of the image by removing and merging those flat zones that do not fulfill the simplification criterion. Thanks to this filtering approach, connected operators cannot introduce any contour distortion related to a specific signal and, as a consequence, they are attractive in a large number of applications where the image has to be simplified without loosing information about contours [5, 9, 11, 14–16, 18]. However, connected operators present a limitation in the way they restore the areas where flat zones have been merged. These areas are always restored as single flat zones that may be perceptible in the filtered image. The basic idea presented in this paper is of removing the perceptual presence of the areas where flat zones have been merged using image inpainting [3, 10].

Image inpainting is to restore missing or inaccessible image pixels in a plausible way based upon the available information. The use of this technique would allows to estimate the areas where flat zones have been merged as natural extension of their surrounding.

The organization of the paper is as follows. The next section provides an introduction to connected operators and analyzes the perceptual effect of the restoring strategy. Section 3 is devoted to image inpainting and a practical algorithm is described. In Section 4, the proposed technique is presented and its structure is discussed. In Section 5, several examples are reported and performances are evaluated. Finally, Section 6 is devoted to the conclusions.

2. Connected operators

2.1 Construction strategies

Connected operators are morphological filtering tools that can simplify part of the image content, while preserving the contours of the remaining parts of the image. This property is a direct consequence of their definition: gray level connected operators filter the image by removing and merging those flat zones that do not fulfill a given simplification criterion. The most successful strategies to construct connected operators rely on a reconstruction process or on a region-tree pruning. The reconstruction process [5,11,17,18] is called anti-extensive, extensive or self-dual, depending on the image components it allows to simplify. Anti-extensive and extensive reconstruction processes deal with either bright or dark image components respectively, whereas self-dual reconstruction deals with all components in a symmetrical way. The self-dual reconstruction [11], also called *leveling*, is defined as follows: if u and v are two images (respectively called the *reference* and the *marker* image), the self-dual reconstruction ρ of v with reference u is defined by: $\rho(v|u) = \lim_{n\to\infty} v_n$ and $v_n = \epsilon_0(v_{n-1}) \bigvee [\delta_0(v_{n-1}) \bigwedge u]$, where ϵ_0 and δ_0 represent respectively an erosion and a dilation with square or a cross of 3×3, $\vee$ and $\wedge$ the infimum and supremum and $v_0 = v$.

An example of self-dual reconstruction is shown in Figure 1. In this example, the marker image is constant everywhere except for two points that mark a maximum and a minimum of the reference image. After reconstruction, the output has only one maximum and one minimum and their contours coincide with those of the reference signal.

In practice, useful connected operators are obtained by considering that the marker v is a transformation $\Phi(u)$ of the input image. The transformation Φ defines the simplification effect and the reconstruction process restores the contour of the flat zones that have not been completely removed by the simplification. As a result, most connected operators obtained by reconstruction can be written as: $\psi(u) = \rho(\Phi(u) \mid u)$, where ρ represents a generic reconstruction process. Examples of filtering effect include size

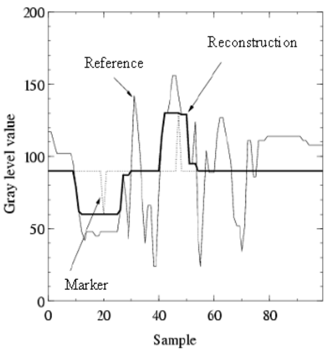


Figure 1. Example of self-dual reconstruction.

oriented (resp. a contrast-oriented) simplification if Φ is an erosion or a dilation with a structuring element (resp. a subtraction or an addition of a constant gray level value).

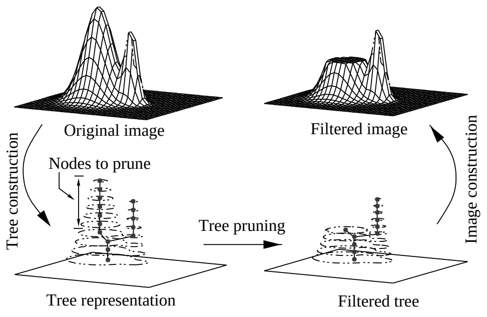


Figure 2. Max-tree processing of images.

Region-tree pruning [15, 16] strategies offer an alternative way to create connected operators. The general approach involves three steps (Figure 2). The first step consists in creating the tree representation of the image: the tree root represents the entire support of the image, the tree nodes represent image regions and the tree branches represent the inclusion relationships among the nodes. The second step is the filtering itself, which analyzes each node and takes a decision on which node has to be preserved and which has to be removed. Finally, the last step restores the filtered image by transforming the pruned tree into a gray level image. Classical trees suitable for connected operators are the Max-tree/Min-tree [16], the Binary Partition Tree (BPT) [15] and the Component Tree [12]. The nodes of a Max-tree (Min-tree) represent the connected components of level sets an image is made of, and are obtained iteratively by thresholding the image at all possible gray levels. The nodes of a BPT represent the regions that can be obtained from an initial partition by merging neighboring regions following an homogeneity criterion until the tree root is obtained. The nodes of a component tree represent connected components of upper- or lower level sets with their holes filled, also called "shapes", which are obtained putting togheter bounded connected components of both upper and lower level sets. The interest of the tree representation is in that the tree structure is fixed and represented by the tree branches. As a consequence, sophisticated pruning strategies can be designed allowing to deal in particular with non-increasing criteria. Formally, a criterion $\mathtt{C}$ assessed on a region R is said to be increasing if the following property holds: $\forall R_1 \subseteq R_2, \mathtt{C}(R_1) \subseteq \mathtt{C}(R_2)$. In the increasing case, there is a relation between the criterion value of a node and that of its descendants in the sense that if a node has to be removed all its descendants have also to be removed. In the case of non-increasing criteria, this relation does not hold and this fact implies a lack of robustness of the operator. Some practical rules have been reported in the literature to deal with the non-increasing case. One strategy is to formulate the problem as a dynamic programming issue and to solve it with the Viterbi algorithm [15, 16].

Summarizing, the region-tree pruning approach is conceptually more complex than the reconstruction approach. However, it leads to a very efficient implementation of connected operators. It allows dealing with non-increasing criteria and provides more flexibility in the choice of the simplification criterion.

2.2 Perceptual analysis

Even if connected operators allow image simplification without introducing distortion on the remaining contours, they may not perceptually remove the presence of areas where flat zones have been merged. Figure 3 illustrates this effect with a size-oriented connected operator: $\psi(u) = \rho(\epsilon(u) \mid u)$, where ϵ is an erosion with a square structuring element of size 5×5 (over a 512×512 image). This operator is known as the opening by reconstruction of erosion [18]. Its goal is to remove small maxima.

Flat zones corresponding to the writing "MPEG4 WORLD" have been removed and merged into a single one whose gray level value depends on the surrounding flat zones. However, the region of support, that is the text, is still visible in the filtered image. In some applications as segmentation or editing, it is desirable to remove regions from the original image without leaving any perceptual information about them.

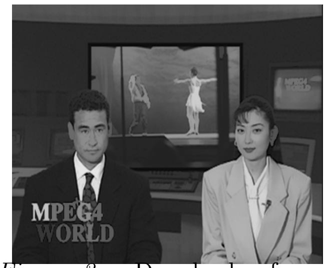


Figure 3. Drawback of connected operators.

In the sequel, the use of inpainting to estimate the pixel values of areas where flat zones have been merged is proposed and discussed.

3. Image inpainting

3.1 A variational approach

In the field of image processing, the term inpainting refers to the task of restoring the pixel values for a missing or consciously masked sub-region of the image domain. The basic idea of the algorithms proposed in the literature [1–4, 6, 7, 10] is to restore the missing regions with the information available from their surrounding. Depending on their goal and on the surrounding information they use, available methods can be broadly classified in *structural* or *textural* inpainting. Structural inpainting [2–4, 10] restores the geometric structures of the image using contour information. Textural inpainting [1, 6, 7] restores the texture of the image using patterns or texture exemplars. Since the use of a connected operator guarantees that at least the contour information is preserved, a structural inpainting algorithm has been used in this work [2]. However, in principle, any state of art inpainting algorithm can be used. The authors formulate the inpainting as the problem of restoring, in the regions of missing data, both the geometric and the photometric information represented respectively by the level lines and the gray level values. Let Ω be the region to be inpainted, and B a narrow band of pixels surrounding Ω. In order to extrapolate the shape information independently from the contrast, the image in $\tilde{\Omega} = (\Omega \cup B)$ is decomposed into level sets U_λ, which are inpainted individually. For each level set U_λ the solution on Ω is obtained minimizing the following functional over $\tilde{\Omega}$:

$$E_\lambda = \int\limits_{\tilde{\Omega}} |div(\theta)|(a + b|\nabla U_\lambda|)dx + \alpha \int\limits_{\tilde{\Omega}} (|\nabla U_\lambda| - \theta \cdot \nabla U_\lambda)dx, \qquad (1)$$

where a, b and α are positive constants and ideally θ represents the normal vector field to the level lines of U_λ, that is: $\theta = \frac{\nabla U_\lambda}{|\nabla U_\lambda|}$. This constraint is expressed with the second integral term. The level set extrapolation is a function of the geometric quantities that appear in the functional over $\tilde{\Omega}$: the curvature of level lines and the perimeter of the discontinuities represented respectively by the quantities $div(\theta)$ and $(a + b|\nabla U_\lambda|)$ in the first integral term. Finally, the solution is obtained by stacking the extrapolated level sets.

3.2 Optimization algorithm

Numerically, the minimization of the functional (1) is computed solving the variational problem via an iterative algorithm based on the gradient descent flow. Since the iteration number depends on the size of Ω and on the homogeneity of the level lines in B, a termination criterion has to be fixed. However, this aspect has not been addressed by the authors and is often neglected in the literature. The solution used in this work has been of considering the variation of the energy as a function of the iteration number and of fixing the termination criterion by a threshold on its slope. In practice, when the local slope of the functional variation is sufficiently close to 0, it is assumed that the algorithm has converged.

4. Proposed approach

In this section, a new filtering strategy allowing to overcome the drawback of connected operators is proposed. The filtering process (Figure 4) involves two major steps: simplification by a connected operator and restitution of the perceptually most important removed flat zones by inpainting. The intermediate step consists in computing the mask marking the regions to be inpainted. The question that may arise now is to know which flat zones, among the new ones created by the filtering process, need to be estimated by inpainting since their are perceptible. The method used in this work is as follows. First, the residue I_3 is binarized with threshold 1. Second, the mask obtained in this way is simplified by removing the connected components for which the residue remains smaller than a fixed threshold. The value of this second threshold guarantees the visibility of the simplification effect. In fact, the intuition behind this solution is that there exists a relationship between the perceptibility of the new flat zones and the visibility of the action of the connected operator. Over the mask obtained after the binarization at two levels (M_1), a closing is applied to merge regions which are very closed to each other (M_2). This operation guarantees that the inpainting algorithm could dispose of a band of at least tree pixels in order to compute geometric features. When the boundary of a flat zone to be inpainted corresponds to an object boundary in the original image, it is often surrounded by a transition zone of at least one or two pixels. This transition zone may disturb the process of capturing the geometry. To avoid this problem, before the inpainting, the regions of the filtered image marked by the complementary of the mask M_2^C are simplified using an opening by reconstruction of erosion (structuring element of size 3×3 or 5×5), followed by its dual, the closing by reconstruction of dilation.

After inpainting, both the simplified version of the band and the inpainted regions marked by the mask are inserted in the image I_2. In order to copy also the simplified band, before of applying the toggle mapping, the mask M_2 is dilated of a size equal to the size of the band used by inpainting. This strategy guarantees that inpainted areas will be perceived as a smooth extension of the visual information contained in the band.

Figure 5 shows an example of the result obtained using the proposed approach. First, the original image I_1 is filtered using a size-oriented connected operator (Figure 5(a)). Second, a mask, defining the perceptually most important regions that have been removed by the connected operator, is computed and the regions that are very close to each other are merged by a closing. The resulting mask M_2 is shown in Figure 5(b). Third, the

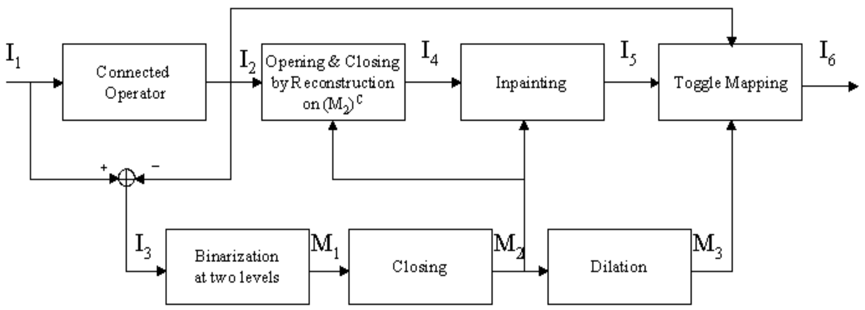
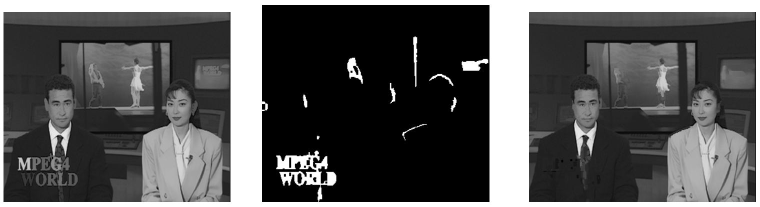


Figure 4. Proposed filtering approach. I refers to an image.; M refers to a mask; M^C refers to the complement of M.



(a) Size operator (b) Inpainting mask (c) Proposed approach

Figure 5. Example of filtering with the size criterion.

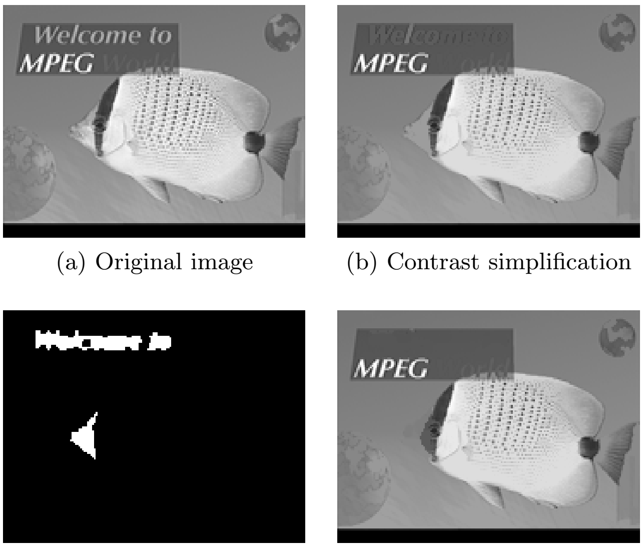
filtered image I_2 is smoothed in the areas surrounding the regions marked by the mask using an opening by reconstruction followed by its dual, the closing by reconstruction. Fourth, the regions of this smoothed image I_4 marked by the mask M_2 are inpainted. Finally, a toggle mapping copies the regions estimated by inpainting, as well as their smooth bands in the image I_2. As can be noticed, the connected components marked by the mask (Figure 5(b)), as for instance the writing and the legs of the dancer, visible in Figure 5(a), are no longer perceived in Figure 5(c), since they have been completely removed by inpainting.

5. Experimental results

5.1 Perceptual filtering examples

This section presents some results obtained using the proposed filtering approach. A first example considering size simplification has been shown in Section 4. The second example involves a contrast simplification.

The contrast simplification is obtained using a λ-max operator: $\psi(u) = \rho(u - c \mid u)$. This operator allows to remove all maxima (or minima) with contrast inferior to λ. In this example, the effect of the contrast simplification (with $\lambda = 100$) is specially visible in areas such as the writing "Wel-



(a) Original image (b) Contrast simplification

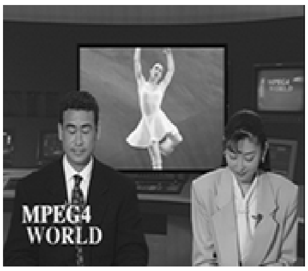
(c) Inpainting mask (d) Proposed approach

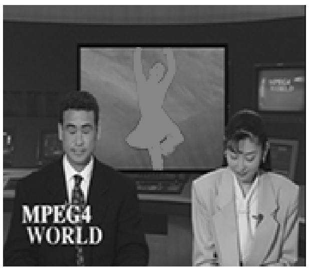
Figure 6. Example of filtering with contrast criterion.

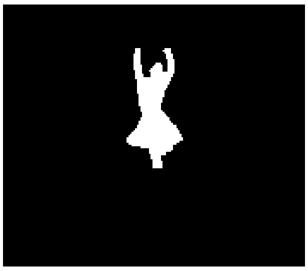
come to" and the muzzle of the fish (Figure 6(b)). However, these regions are still perceptible. Instead, applying inpainting to the regions marked by the mask (Figure 6(c)), the perceptual presence of these areas is removed (Figure 6(d)).

The third example involves a motion criterion in image sequences [8]. Motion operator allows to remove from an image all objects that do not undergo a given motion. In this example, the operator has been applied to remove all moving objects from the image shown in Figure 7(a). In the considered sequence, the perceptually most important moving object is the dancer behind the two speaker. As can be observed in Figure 7(b), although flat zones corresponding to the dancer have been removed, they appear in the filtered image as a large flat zone inside a smoothed region. Using inpainting to estimate pixel values in this area (Figure 7(c)), the perception of this flat zone is completely removed (Figure 7(d)).

The last example involves an image obtained superimposing white lines to the original image. The goal of this example is to use the original image as reference to perform an objective quality assessment. Assume that the goal is to remove the superimposed lines from the image shown in Figure 9(a). Due to their shape, the white lines are suited to be removed using a complexity criterion [13]. This criterion is based on a measure of the ratio between the perimeter P and the area A of the connected component. Intuitively, if a connected component has a small area but a very long perimeter, it corresponds to a complex object. However, a complexity criterion alone would remove also a large number of complex flat zones.


(a) Original image


(b) Motion simplification


(c) Inpainting mask

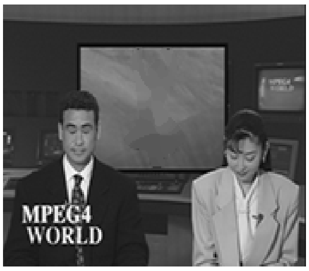
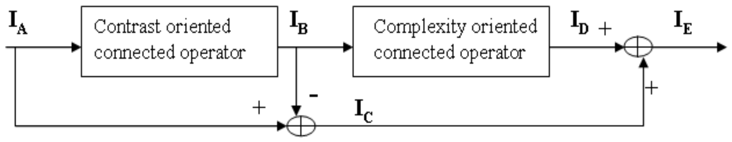

(d) Proposed approach

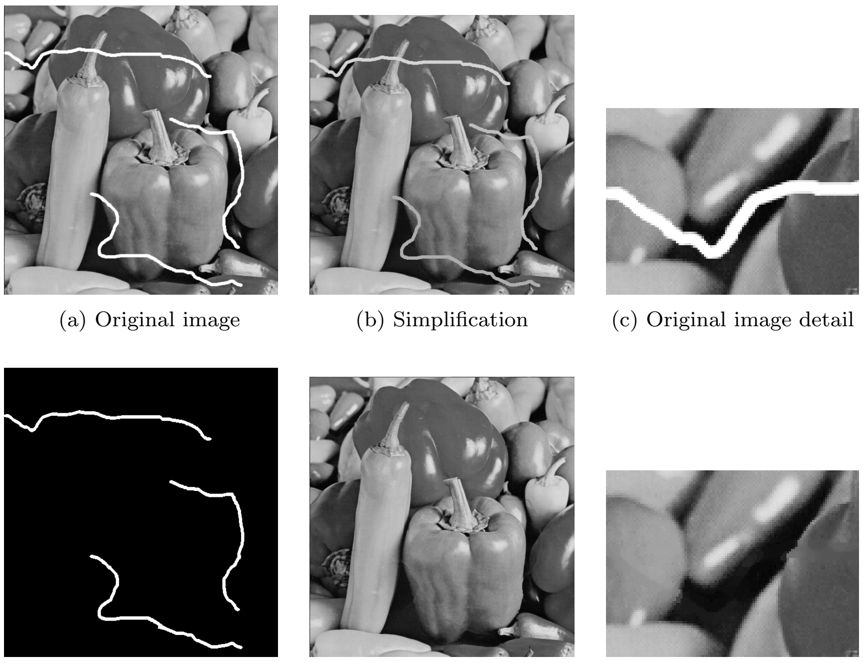
Figure 7. Example of filtering with a motion criterion.



I_A
Contrast oriented connected operator
I_B
Complexity oriented connected operator
I_D +
I_E
+
-
I_C
+

Figure 8. Connected operator used to extract the white lines.

In order to extract the white lines removing the smallest number of other flat zones, the connected operator shown in Figure 8 has been used. This operator is based on the observation that the white lines are complex and highly contrasted, whereas other complex flat zones correspond to texture which is less visible because of low contrast. First, the image shown in Figure 9(a) is filtered using a contrast-oriented connected operator, which removes all maxima having contrast inferior to 170. This is the maximum value of the contrast allowing to preserve the white lines. Second, the resulting image is filtered by a complexity criterion, with complexity 42, that is the minimum value allowing to remove the white lines. Note that the complexity criterion is not increasing because there is not a relationship of complexity between two connected components R_1, R_2 such that $R_1 \subset R_2$. In order to deal with this non increasing criterion the "Max" decision has been employed. The "Max" decision is defined as follows: a node C for which the evaluation of the criterion is lower than a given threshold is not removed if at least one of its descendants has to be preserved. Finally, the difference I_C between the original image I_A and the image filtered using a contrast criterion I_B is added to the filtered image using a complexity criterion I_D to restore the texture.


(a) Original image (b) Simplification (c) Original image detail

(d) Inpainting mask (e) Proposed approach (f) Result image detail

Figure 9. Example of filtering with the connected operator of Figure 8.

As can be observed in Figure 9(b), flat zones corresponding to the white lines have been removed and most of the texture preserved. However, the perceptual presence of the superimposed lines is still strongly visible. Instead, applying inpainting to the perceptual more important removed regions (Figure 9(d)) a much better result is obtained (Figure 9(e)).

5.2 Performances evaluation

In this section the quality performances of the proposed approach are compared objectively to classical filtering strategies using two well defined criteria: the peak signal-to-noise ratio (PSNR) and the Structural Similarity Image Measure (SSIM) [19,20].

The PSNR relies on the mean square error (MSE) which is the square of the Euclidean distance between a reference and a distorted image. This definition does not include any perceptual features and therefore MSE and PSNR do not really assess the perceived image quality. To evaluate the

Table 1. PSNR and SSIM values obtained using different filtering approaches.

Type of filter	*PSNR value*	*SSIM value*
Low pass filter – average (11 × 11) (5 × 5)	19.8	0.06
Median filter (15 × 15) (5 × 5)	23.2	0.87
Connected operator	21.9	0.91
Proposed approach	39.2	0.99

perceived image quality, the SSIM has been used. The SSIM relies on the assumption that human vision is highly sensitive to structural information and, as consequence, a measure of the structural information change should provide a good approximation of perceived image distortion. In practice, the SSIM quantifies the differences between a distorted image and a reference image independently from average luminance and contrast. The SSIM is equal to 1 for two identical images. Performances have been evaluated for the last example described in Section 5, for which the reference image (image without the superimposed lines) is available. Table 1 shows the PSRN values as well as the SSIM values obtained using different filtering techniques. In the case of median and low pass filter, the table reports the best value obtained using different sizes of window ranging from 5×5 to 25×25. As can be observed, the proposed scheme drastically increases both the PSNR and the SSIM, meaning that the approach proposed in this paper reduces the error visibility and gives results consistent with the qualitative visual appearance.

The proposed filtering strategy is computationally more complex than connected operators, depending its complexity on the inpainting technique used. However, it permits to achieve a good trade-off between quality result and computational cost.

6. Conclusions

In this paper, a new filtering technique improving connected operator performances has been presented and discussed. The proposed technique involves two broad steps: first, the image is simplified using connected operators. Second, the perceptually most important filtered regions are estimated using inpainting. The mask marking the regions to be inpainted is automatically computed and no user interaction is required. Comparative experiments have shown that the proposed technique outperforms classical filtering strategies in terms of both visibility of error (PSNR) and structural perceptual quality (SSIM). The presented approach is general in the sense that any connected operator can be used. As a result, it is suitable for a large set of advanced filtering applications such as objects, writing or defects removal. The future work will be devoted to improve the current method of selecting the regions to be inpainted.

References

[1] M. Ashikhmin, *Synthesizing natural textures*, Proc. ACM Symposium on Interactive 3D Graphics (2001), 217–226.

[2] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, and J.Verdera, *Filling-in by joint interpolation of vector fields and grey levels*, IEEE Trans. on Image Processing **10** (2001), no. 8, 1200–1211.

[3] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, *Image Inpainting*, Proc. 27th Annual Conf.on Computer Graphics and Interactive Techniques, 2000, pp. 417–424.

[4] T. F. Chan and J. Shen, *Mathematical models for local nontexture inpaintings*, SIAM Journal of Applied Mathematics **62** (2001), no. 3, 1019–1043.

[5] J. Crespo, J. Serra, and R. W. Schafer, *Theoretical aspects of morphological filters by reconstruction*, Signal Process **47(2)** (1995), 201–225.

[6] A. Efros and W. T. Freeman, *Texture synthesis by non-parametric sampling*, Proc. of ACM Conf. on Computer Graphics, SIGGRAPH (2001), 341–346.

[7] A. Efros and T. Leung, *Texture Synthesis by non-parametric sampling*, Proc. of Int. Conf. on Computer Vision, ICCV (1999), 21033–1038.

[8] L. Garrido, A. Oliveras, and P. Salembier, *Motion analysis of image sequences using connected operators*, Proc. of Visual Communications and Image Processing, SPIE, 1997, pp. 546–557.

[9] H. J. A. M. Heijmans, *Connected Morphological Operators and filters for binary images*, Proc. of Int. Conference on Image Processing, ICIP, 1997, pp. 211–214.

[10] S. Masnou and J.-M. Morel, *Level-lines based disocclusion*, Proc. of Int. Conf. on Image Processing, ICIP, 1998, pp. 259–263.

[11] F. Meyer, *The levelings*, Proc. of Int.Symposium on Mathematical Morphology, ISMM, 1997, pp. 211–214.

[12] P. Monasse and F. Guichard, *Scale-Space from a Level Lines Tree*, Journal of Visual Com. and Image Rep. **11** (2000), 224–236.

[13] A. Oliveras and P. Salembier, *Generalized connected operators*, Proc. of Visual Communications and Image Processing, SPIE, 1996, pp. 761–772.

[14] G. K. Ouzounis and M. H. F. Wilkinson, *Second order attribute filters using Max-Trees*, Proc. of Int. Symposium on Mathematical Morphology, ISMM, 2005, pp. 65–74.

[15] P. Salembier and L. Garrido, *Binary partition tree as an efficient representation for image processing, segmentation and information retrieval*, IEEE Trans. on Image Processing **7(4)** (2000), 561–576.

[16] P. Salembier, A. Oliveras, and L. Garrido, *Anti-extensive connected operators for image and sequence processing*, IEEE Trans. on Image Processing **7(4)** (1998), 555–570.

[17] P. Salembier and J. Serra, *Flat zones filtering, connected operators and filters by reconstruction*, IEEE Trans. on Image Processing **3(8)** (1995), 1153–1160.

[18] L. Vincent, *Morphological gray scale reconstruction in image analysis: Applications and efficient algorithms*, IEEE Trans. on Image Processing **2(2)** (1993), 176–201.

[19] Z. Wang, *The SSIM index for Image Quality Assessment.* MATLAB implementation available online from: http://www.cns.nyu.edu/∼lcv/ssim/.

[20] Z. Wang, A. C. Bovik, and E. P. Simoncelli, *Image Quality Assessment: from Error Visibility to Structural Similarity*, IEEE Trans. on Image Processing **13(4)** (2004), 600–612.

Segmentation using vector-attribute filters: methodology and application to dermatological imaging

Benoît Naegel[1,2], Nicolas Passat[3], Nicolas Boch[4] and Michel Kocher[1]

[1] EIG-HES, Geneva School of Engineering, Geneva, Switzerland
{benoit.naegel,michel.kocher}@hesge.ch

[2] LORIA, Campus Scientifique, Vandœuvre-les-Nancy, France

[3] LSIIT, UMR 7005 CNRS/ULP, Strasbourg 1 University, France
passat@dpt-info.u-strasbg.fr

[4] Digital Imaging Unit, Department of Radiology, Geneva University Hospital, Switzerland

Abstract Attribute-based filters can be involved in analysis and processing of images by considering attributes of various kinds (quantitative, qualitative, structural). Despite their potential usefulness, they are quite infrequently considered in the development of real applications. A cause of this underuse is probably the difficulty to determine correct parameters for non-scalar attributes in a fast and efficient fashion. This paper proposes a general definition of vector-attribute filters for grey-level images and describes some solutions to perform detection tasks using vector-attributes and parameters determined from a learning set. Based on these elements, an interactive segmentation method for dermatological application has been developed.

Keywords: vector-attribute filters, component-tree, segmentation, dermatological imaging.

1. Introduction

Connected operators are fundamental tools of mathematical morphology: area filters [20, 21], contrast filters [5], or volumic filters [19] are all connected operators that, given some criteria, simplify the partition of an image without introducing new contours. Attribute openings and thinnings have been introduced in [1] and generalise the notion of connected filters based on arbitrary attributes.

These filters can be efficiently implemented using a tree structure known in the literature as *dendrone* [2,6], *component-tree* [10], *confinement tree* [8] or *max-tree* [13]. In the sequel we will employ the generic term *component-tree* to denote all these tree-like structures. Their main principle consists in storing each connected component of the successive threshold sets of a grey-level image in a node, and to code the inclusion relation on components by establishing links between the corresponding nodes. To each node of the tree can be associated a real-valued attribute, leading to an efficient tool to design connected, anti-extensive filters [13]. Recently, multi-valued attributes have been considered leading to vector-attribute filters [16].

Such filters seem to have an interesting potential for real application development. However, their main drawback is that attribute parameters are still mainly determined in an empirical fashion, as in [7] where the classification between an object component and the background is made by observation of the attribute signature of this component.

Until now, attribute-filters have been essentially used for removing components presenting undesirable attributes. We wish to demonstrate here that attribute-filters, and more especially vector-attribute filters, can also be efficiently involved in the design of segmentation methods.

This paper is organised as follows. Section 2 presents recent works related to the development of attribute-based filters and their use for applicative purpose. In Section 3 we propose a definition of vector-attribute filters for grey-level images and propose to use them in an object-detection context. Section 4 describes an applicative study of the proposed methodology devoted to the interactive analysis and segmentation of melanocytic nevi in 2D dermatological images. In Section 5 possible developments, improvements and further works are discussed.

2. Related work

Attribute-based filtering using component-trees has been used in various contexts, including feature extraction and retrieval [2], image coding and compression [13], or segmentation of vessels in wood micrographs [7]. Component-trees have also been used in the field of cerebral segmentation, for the automatic selection of markers from 3D MRI images [3].

The notion of shape-based attributes has been considered in [16–18] as well as the notion of shape granulometry. These concepts have been applied to filament extraction in MR angiograms [18] and classification of diatoms [17].

Component-tree representation of an image based on a connectivity map (leading to a second-order connectivity) has been proposed in [11, 12] and applied to 3D filament extraction in MR angiograms and extraction of filamentous structures in images of proteins.

Vector-attribute filters have been introduced recently in [16], where their use was illustrated on synthetic images of characters, in the context of object filtering based on Hu's moments invariants. Although vector-attribute based filters seem to have a great potential, they have not been used until now in concrete applications.

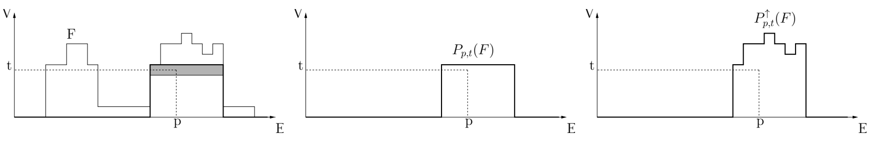


Figure 1. Decomposition of a function into its peak components. Left: Original function F. In grey: connected component of $X_t(F)$ including p. Middle: Peak component $P_{p,t}(F)$. Right: Lobe component $P^{\uparrow}_{p,t}(F)$.

3. Grey-level vector-attribute filters

The connected components of all threshold sets of an image are usually called *peaks*. In the context of object recognition or segmentation, it could be interesting to detect from a grey-level image the peaks representing a specific structure: only these peaks will be preserved, while all other peaks will be removed.

Formally, a grey-level image is defined as a numerical function $F : E \to V$, where E is a space of points and V a totally ordered set of values, where $\perp$ (resp. $\top$) represents the least (resp. the greatest) element.

The threshold set of a function is defined by $X_t(F) = \{p \in E \mid F(p) \geq t\}$. Given a connectivity class $\mathcal{C}$ on E (i.e., the set of all connected sets), the connected opening for sets is defined as [15]: $\gamma_x(X) = \bigcup\{C \mid x \in C \subseteq X, C \in \mathcal{C}\}$.

We define the peak function $P_{p,t}(F)$ by:

$$P_{p,t}(F)(x) = \begin{cases} t & \text{if } x \in \gamma_p(X_t(F)), \\ \perp & \text{otherwise.} \end{cases}$$

Any function F is the supremum of all its peak components:

$$F = \bigvee\{P_{p,t}(F) \mid p \in E, t \in V\}.$$

We define the lobe (or superior peak), as:

$$P^{\uparrow}_{p,t}(F)(x) = \begin{cases} F(x) & \text{if } x \in \gamma_p(X_t(F)), \\ \perp & \text{otherwise.} \end{cases}$$

These functions are illustrated in Figure 1.

Given a valuation function $\tau : V^E \to \mathbb{R}^N$ that associates to a function F a vector of real-valued attributes $\tau(F)$, and a criterion $T : \mathbb{R}^N \to \{true, false\}$ that accepts or rejects an attribute-vector depending on a particular strategy, a grey-level vector-attribute filter φ can be defined by acting separately on the peak components of an image:

$$\varphi(F) = \bigvee\{P_{p,t}(F) \mid p \in E, t \in V, T(\tau(P^{\uparrow}_{p,t}(F))) = true\}. \qquad (1)$$

The reconstruction of the lobes can be obtained in the same way:

$$\varphi^{\uparrow}(F) = \bigvee\{P^{\uparrow}_{p,t}(F) \mid p \in E, t \in V, T(\tau(P^{\uparrow}_{p,t}(F))) = true\}. \qquad (2)$$

In the case where all peak functions are removed, the result of these filtering operations is the least element of the complete lattice of functions V^E: $\bigvee \emptyset = C_\perp$ (where $C_\perp$ is the constant function defined by: $\forall p \in E, C_\perp(p) = \perp$). These filters associate to each lobes of a function a vector of attributes. Using lobes allows to consider non-flat attributes like contrast (i.e., height), or volume of the peak component. If one of the vector-attribute component is non-flat, φ is no longer idempotent since it reconstructs only the peak $P_{p,t}$[1]. On the contrary, $\varphi^{\uparrow}$ is idempotent, since the lobes verifying the criterion are preserved. These filters are not increasing most of the time, since the criterion based on a vector-attribute is seldom increasing. Hence, they are not *morphological* filters. They are anti-extensive, as they remove peaks from the original function. Finally, as they act only by merging image flat-zones, they are connected operators.

3.1 Object detection

The filters previously described can be used to perform object detection from a grey-level image provided that objects of interest correspond to some peaks of the image: this requires the object to be a bright structure surrounded by dark background (at least after some kind of preprocessing). Such a strategy requires one to have some prior knowledge about the object to segment.

A typical example of criterion (also proposed in [16]) is based on a distance d from a reference vector $\mathbf{r}$:

$$T_{\mathbf{r},\varepsilon}(\mathbf{v}) = \begin{cases} true & \text{if } d(\mathbf{r},\mathbf{v}) < \varepsilon, \\ false & \text{otherwise.} \end{cases}$$

In this latter case, note the difference with the vector-attribute defined in [16]: here peaks are suppressed when their attribute-vectors have a distance superior to ε. In [16], the opposite was performed in order to remove sets having attributes close to the reference vector.

3.2 Vector-attribute filtering using component-tree

Vector-attribute filters, as defined previously, can be efficiently implemented using the image component-tree. Moreover, as we will see in the sequel, the component-tree only needs to be computed once, allowing one to perform multiple consecutive filtering with different parameters $\mathbf{r}$ and ε. The

[1]For the same reason, h-reconstruction or volumic filters are not idempotent and hence are not morphological filters.

component-tree of a function F can be defined as follows. Each node of the tree is a peak function belonging to the set: $P(F) = \{P_{p,t}(F) \mid p \in E, t \in V\}$. For $G : E \to V$, we consider the function:

$$supp(G) = \begin{cases} \{p \in E \mid G(p) > \bot\} & \text{if } G \neq C_\bot, \\ E & \text{if } G = C_\bot. \end{cases}$$

A node P_2 is a child of P_1 in the component-tree of F ($P_1, P_2 \in P(F)$) iff:

(i) $supp(P_2) \subset supp(P_1)$,

(ii) $\forall P_3 \in P(F), supp(P_2) \subset supp(P_3) \Rightarrow supp(P_1) \subseteq supp(P_3)$.

The root of the component-tree of F is the node $R \in P(F)$ such that $supp(R) = \bigcup_{P \in P(F)} \{supp(P)\} = E$. In particular, $R = C_{F_{min}}$, where $F_{min} = \min\{F(p) \mid p \in E\}$.

Algorithmically, each node can be modelled as a structure : $node = (label, gl,$
$size, att, points, parent, children, active)$, where:

- *label* is the identifier of the node;
- *gl* is the grey-level of the node ($gl(P) = \max\{F(p) \mid p \in supp(P)\}$);
- *size* is the size of the node ($size(P) = card(supp(P))$;
- *att* is a list of attributes, representing the attribute-vector attached to the node;
- *points* is the list of points belonging to the node;
- *parent* is a pointer to the parent;
- *children* is a list of pointers to the node's children;
- *active* is a Boolean value indicating the status of the node.

It is desirable to exploit the redundancy of the points belonging to the support of the peaks: each point p can be stored in only one node ($P_{p,F(p)}(F)$). This is the principle adopted in [13], leading to the max-tree. In this case, some nodes have no points: they can be suppressed, leading to the unique representation of the tree [8]. The construction of the component-tree can be done using efficient algorithms [8, 10, 13].

Given a valuation function τ and a criterion T, an image filtered by applying Equation 1 can be processed by computing the component-tree of the original image and reconstruct only the nodes verifying T: this is equivalent to prune the tree using the *direct* decision [13], as illustrated in Figure 2. A tree-based filtering can be decomposed into three steps: component-tree computation; tree filtering; image restitution using the *direct* strategy.

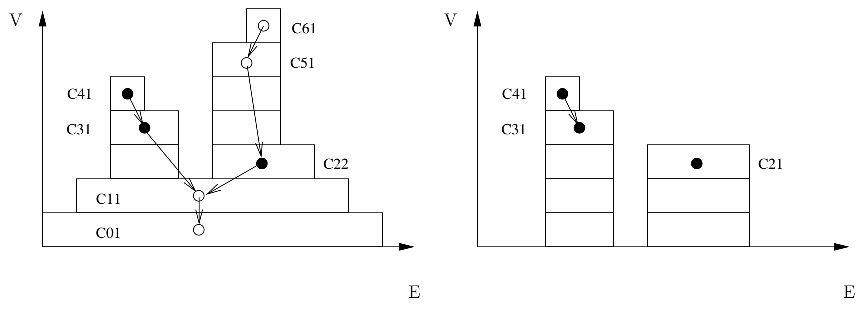


Figure 2. Left: Original image. Empty circles denote nodes that do not meet a considered criterion. Right: Direct reconstruction of peak functions that meet this criterion (applying equation 1).

3.3 Attributes

To each lobe of a function is attached a vector of attributes generated by the τ function. Vector-attributes can represent any combination of photometric, textural, or geometric attributes. The attributes most usually considered for the design of component-tree methods are intensity, area, height (or contrast), and volume, the last two one being defined by:

- Height (or contrast): $height(P^{\uparrow}) = \max_{p \in supp(P^{\uparrow})}\{F(p) - gl(P^{\uparrow}) + 1\}$,
- Volume: $volume(P^{\uparrow}) = \sum_{p \in supp(P^{\uparrow})}\{F(p) - gl(P^{\uparrow}) + 1\}$.

Geometric or shape attributes have an interesting potential since they enable to discriminate an object by considering its structural properties, by opposition to more classical photometric properties. In [22], various shape representation techniques are described, which could lead to the computation of such attributes. However when using the component-tree, it is desirable (for efficiency reasons) to use attributes that can be computed incrementally during its construction.

3.4 Attribute learning from a reference component-tree

Using vector-attribute filters for object detection requires a preliminary characterisation of a specific class of objects. A possible way to retrieve some information regarding the attributes of a class is to use a set of object samples (or learning set). This learning set can be composed, for example, of manually delineated structures on some original images. Given images F_i in which the structures of interest have been segmented (each structure being a connected set S_i), it is possible to extract, for each manually segmented component, the most corresponding peak of F_i. This can be done using the component-tree of F_i: to each point of the manual segmentation $p \in S_i$ is associated the corresponding node or peak function $P_{p,F(p)}(F)$. Algorithmically, this correspondence can be carried out using a mapping between the points and their corresponding nodes. To each component S_i is associated a set of potential peaks. A way to process this can consist in retrieving the node associated to the peak having the closest size to the component's size (i.e., $card(S_i)$).

The attribute-vector of the node can then be retrieved and associated to the component S_i. The set of attribute-vectors corresponding to the manually segmented structures represent the learning set of the structure of interest.

4. Application: Segmentation of melanocytic nevi from photographs of the whole body

Early detection of skin cancer is a very important issue to prevent the mortality due to this kind of affection. Computer-aided diagnosis represents one step towards a more accurate and faster detection of suspicious moles. Much of the research effort in this domain has been done in the field of automated diagnosis from dermatoscopy examinations [4,14]. However this kind of methods requires the dermatologist to detect beforehand the suspicious lesions, which is difficult due to the large number of moles in the patients at risk.

Mole mapping from digital photographs is a relatively recent method whose purpose is to assist the dermatologist in the detection of suspicious moles. To this aim, a cartography of the existing moles is performed from images of the whole body acquired at different times. Changes can be tracked by comparing automatically the corresponding moles in the different images.

In previous work, a mole-mapping system has been proposed [9], enabling the detection of the moles in several images of a patient and the matching of the corresponding structures. Although based on empirically evaluated parameters, the method was automatic and efficient due to the component-tree approach. We propose here an improved method which allows the dermatologist to segment interactively a class of moles of interest (for example only the largest ones) and to use the specific parameters of this class to infer the segmentation of the closest moles belonging to the same one.

4.1 Definitions and notations

Photographs of the whole body are considered as functions $F : E \to \mathcal{T}^{rgb}$ where $\mathcal{T}^{rgb} = \mathcal{T}^r \times \mathcal{T}^g \times \mathcal{T}^b$ represents the set of colour values defined by a triplet (r, g, b). Images are processed in the saturation space to discriminate moles from skin: indeed, moles and skin have similar hue, but moles appear more saturated than the skin. The saturation image is defined by the function: $S : E \to \mathcal{T}^s : x \mapsto \frac{max-min}{max}$ where max (resp. min) represents the largest (resp. the smallest) value of the RGB triplet $F(x) = (r, g, b)$. We define $S(x) = 0$ for $F(x) = (0, 0, 0)$. An original image, and a sample visualised in the saturation space, are illustrated in Figure 3.

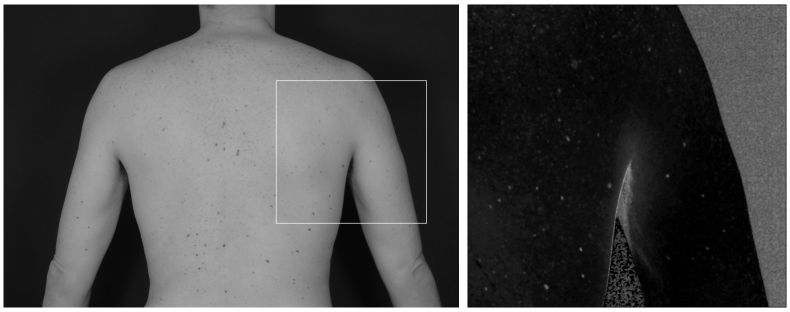


Figure 3. Left: Original image F (visualised in grey-levels, but defined in the RGB colour space). Right: Enlargement of the white-bordered zone, in the saturation image S associated to F.

4.2 Interactive segmentation algorithm

Photographs of the whole body consist in a total of 16 images of a patient in the front, back, right and left positions. The acquisition of each image set requires a calibration of the digital camera. Dimensions of each image are 4288×2848 (12 megapixels). The corresponding physical size of the image pixels obtained from the calibration step is typically contained between 0.16 and 0.17 mm. Individual images of the corresponding part of the body are then selected by the practitioner for the examination.

Input and output The segmentation algorithm takes as input two digital photographs of similar parts of the body acquired at different times. The practitioner can interactively contour the moles of interest, providing a first manual segmentation. A minimum of three or four segmentations is required to obtain sufficient data for the computation of a statistical model. A distance parameter can be chosen interactively. As output, the algorithm detects the moles which are the closest from the manually selected ones, given the chosen distance parameter. The mole segmentation in each of the image can then be used as the input of a specific point matching algorithm (not described here) [9].

Attributes and criterion In the saturation images, moles appear as bright and compact structures. Cutaneous surfaces have uneven illumination: segmentation methods based on global threshold are not sufficient to discriminate moles from other bright structures (see Figure 3 right, the umbra under the arm appears very bright in the saturation image). To each lobe $P^{\uparrow}(S)$ of the saturation image can be associated a vector-attribute composed of area, contrast, and compacity parameters:

$$\tau(P^{\uparrow}(S)) = (area(P^{\uparrow}(S)), contrast(P^{\uparrow}(S)), compacity(P^{\uparrow}(S))).$$

The compacity parameter can be a measure of roundness (for example the ratio $\frac{4\pi A}{P^2}$)), where A and P represent the area and perimeter of the component, respectively. The perimeter can be approximated by using the number of contour points of the component, however this attribute cannot be computed incrementally (it can however easily be computed afterwards using a mapping between the points and the nodes). Hence an alternative is to use the first Hu's invariant moment, that can be computed incrementally as suggested in [17]. The two variants of compacity have been experimentally tested with similar results.

As a criterion, we use $T_{\mathbf{r},\varepsilon}$, in order to suppress peaks that differ of more than a certain quantity ε from a reference vector. The subset of manually segmented moles is used to select, for each connected component, the closest corresponding nodes of the component-tree, using the strategy described in Section 3.4. The set of vector-attributes $\{\mathbf{v_i}\}$ of all selected nodes is used to compute a reference vector:

$$\mathbf{r} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{v_i},$$

and a covariance matrix:

$$\Sigma = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{v_i} - \mathbf{r})^T(\mathbf{v_i} - \mathbf{r}),$$

where N is the number of manually segmented moles.

Segmentation step Assuming that the distribution of the vector-attributes of the moles is Gaussian multivariate, we can consider the statistical model computed previously to use normalised distances. Given the reference vector $\mathbf{r}$ and the covariance matrix Σ, a distance is computed between $\mathbf{r}$ and the vector-attributes $\mathbf{x}$ of all the nodes of the component-tree. This distance expresses the probability that the node belongs to the class selected by the practitioner. This distance can be the Mahalanobis distance:

$$d_M(\mathbf{x}, \mathbf{r}) = \sqrt{(\mathbf{x} - \mathbf{r})^T\Sigma^{-1}(\mathbf{x} - \mathbf{r})}.$$

Hence parameter ε in the criterion $T_{\mathbf{r},\varepsilon}$ defines the sensitivity of the detection. Choosing $\varepsilon = 1.0$ ensures that the selected components are "not farther" than one standard deviation from the reference vector.

4.3 Experiments and results

Experiments have been made on 12 image series. The computation time of the component-tree using Salembier's recursive algorithm[2] is 8 seconds on a Pentium IV 3.2 GHz with 2 Gb of RAM. Each filtering step is made in constant time (0.1 s), allowing an interaction of the dermatologist in real-time.

Some results are illustrated on Figure 4 (for representation purpose, only an enlarged portion of the processed image is shown) for two different sets of manually segmented moles. In the first row the largest moles are detected, while in the second row, only the smallest ones are preserved due to the different training sets. Segmentation is precise since the attribute filter is a connected operator: detected contours correspond to true contours of the detected component. The segmentation method allows a very good discrimination between moles and other bright structures of the saturation image: there is no false detections. This is mainly due to the chosen criteria. Visual assessment of the detected moles by dermatologist tends to prove that the method is very satisfying from a medical point of view. As the chosen attributes are highly uncorrelated in this application, Mahalanobis distance and normalised Euclidean distance give comparable results.

5. Conclusion and further works

In this paper we have proposed a general definition of vector-attribute filters for grey-level images. This definition allows to filter the function peaks given a vector-attribute and a criterion. We have shown that this definition can be efficiently implemented using the component-tree and the *direct* pruning strategy. Some solutions for using vector-attributes and involving them in the development of component-tree-based filtering processes (especially devoted — but not restricted to — segmentation) have been proposed. They constitute an initial and partial methodological framework which will be enriched in further works.

This framework has been used for the proposal of a segmentation method of dermatological 2D data, allowing a real-time interaction with the practitioner. The efficiency of this method is satisfying from a medical points of view, but also from a theoretical one. Indeed, it tends to prove that it is

[2]According to [10], Salembier's algorithm is quadratic in the worst case; however it is generally twice as fast as Najman's one in practical cases when the value of a point is comprised between 0 and 255.

possible to create efficient, fast and easy to use methods for object detection purpose, based on vectorial attribute filters and component-trees.

Further work will now consist in developing this methodology, by integrating more descriptive attributes (for example structural shape descriptors) and extending it to classification tasks.

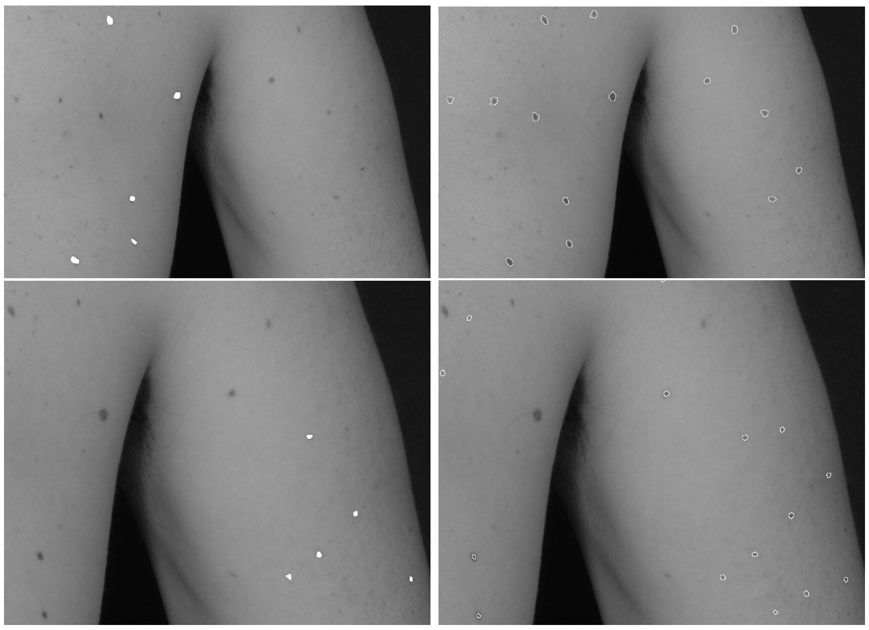


Figure 4. First column: 5 manually segmented moles (in white). Second column: Moles segmented using a normalised Euclidean distance threshold of 3.0. Using compacity as shape attribute allows to efficiently discriminate moles from other bright structures of the saturation image (see Figure 3, right, for comparison).

Acknowledgements

This work has been funded by the Competency Network for Information and Communication Technologies from the University of Applied Sciences of Western Switzerland (HES-SO), under grant # 16198. Images were provided by the dermatological service of Geneva University Hospital.

References

[1] E. J. Breen and R. Jones, *Attribute openings, thinnings, and granulometries*, Computer Vision and Image Understanding **64** (1996), no. 3, 377–389.

[2] L. Chen, M. W. Berry, and W. W. Hargrove, *Using dendronal signatures for feature extraction and retrieval*, International Journal of Imaging Systems and Technology **11** (2000), no. 4, 243–253.

[3] P. Dokládal, I. Bloch, M. Couprie, D. Ruijters, R. Urtasun, and L. Garnero, *Topologically controlled segmentation of 3D magnetic resonance images of the head by using morphological operators*, Pattern Recognition **36** (2003), no. 10, 2463–2478.

[4] S. Fischer, P. Schmid, and J. Guillod, *Analysis of skin lesions with pigmented networks*, Proceedings of the International Conference on Image Processing (ICIP'96), 1996, pp. 323–326.

[5] M. Grimaud, *New measure of contrast: dynamics*, Image Algebra and Morphological Image Processing III, July 1992, pp. 292–305.

[6] P. Hanusse and P. Guillataud, *Sémantique des images par analyse dendronique*, RFIA'91 - Reconnaissance des Formes et Intelligence Artificielle, 1991, pp. 577–588.

[7] R. Jones, *Connected filtering and segmentation using component trees*, Computer Vision and Image Understanding **75** (1999), no. 3, 215–228.

[8] J. Mattes and J. Demongeot, *Efficient algorithms to implement the confinement tree*, DGCI'00 - Discrete Geometry for Computer Imagery, 2000, pp. 392–405.

[9] B. Naegel and N. Boch, *Un système d'aide au diagnostic dermatologique par cartographie et suivi de grains de beauté*, Proceedings of the french conference orasis 2007, 2007.

[10] L. Najman and M. Couprie, *Building the component tree in quasi-linear time*, IEEE Transactions on Image Processing **15** (2006), no. 11, 3531–3539.

[11] G. K. Ouzounis and M. H. F. Wilkinson, *Second-order connected attribute filters using max-trees*, ISMM'05 - International Symposium on Mathematical Morphology, 2005, pp. 65–74.

[12] ———, *Mask-based second-generation connectivity and attribute filters*, IEEE Transactions on Pattern Analysis and Machine Intelligence **29** (2007), no. 6, 990–1004.

[13] P. Salembier, A. Oliveras, and L. Garrido, *Anti-extensive connected operators for image and sequence processing*, IEEE Transactions on Image Processing **7** (1998), no. 4, 555–570.

[14] P. Schmid-Saugeon, J. Guillod, and J. P. Thiran, *Towards a computer-aided diagnosis system for pigmented skin lesions*, Computerized Medical Imaging and Graphics **27** (2003), 65–78.

[15] J. Serra, *Image Analysis and Mathematical Morphology. Vol 2. Theoretical Advances*, Academic Press, London, 1988.

[16] E. R. Urbach, *Vector attribute filters*, ISMM'05 - International Symposium on Mathematical Morphology, 2005, pp. 95–104.

[17] ———, *Connected shape-size pattern spectra for rotation and scale-invariant classification of gray-scale images*, IEEE Transactions on Pattern Analysis and Machine Intelligence **29** (2007), no. 2, 272–285.

[18] E. R. Urbach and M. H. F. Wilkinson, *Shape-only granulometries and gray-scale shape filters*, ISMM'02 - International Symposium on Mathematical Morphology, 2002, pp. 305–314.

[19] C. Vachier, *Utilisation d'un critère volumique pour le filtrage d'image*, RFIA'98 - Reconnaissance des Formes et Intelligence Artificielle, 1998, pp. 307–315.

[20] L. Vincent, *Morphological area openings and closings for grey-scale images*, NATO Shape in Picture Workshop, 1992, pp. 197–208.

[21] ———, *Grayscale area openings and closings, their efficient implementations and applications*, EURASIP Workshop on Mathematical Morphology and its Applications to Signal Processing, 1993, pp. 22–27.

[22] D. Zhang and G. Lu, *Review of shape representation and description techniques*, Pattern Recognition **37** (2004), no. 1, 1–19.

VI

WATERSHED SEGMENTATION

Some links between min-cuts, optimal spanning forests and watersheds

Cédric Allène[1, 2], Jean-Yves Audibert[1], Michel Couprie[2], Jean Cousty[2] and Renaud Keriven[1]

[1] *CERTIS, Ecole des Ponts, Champs-sur-Marne, France*
{allene,audibert,keriven}@certis.enpc.fr

[2] *Université Paris-Est, LABINFO-IGM, UMR CNRS 8049, A2SI-ESIEE, France*
{c.allene,m.couprie,j.cousty}@esiee.fr

Abstract Different optimal structures: minimum cuts, minimum spanning forests and shortest-path forests, have been used as the basis for powerful image segmentation procedures. The well-known notion of watershed also falls into this category. In this paper, we present some new results about the links which exist between these different approaches. Especially, we show that min-cuts coincide with watersheds for some particular weight functions.

Keywords: min-cuts, spanning forests, watershed, shortest-path forest.

Introduction

Min-cuts (graph cuts) and watersheds are two popular tools for image segmentation, which can both be expressed in the framework of graphs and are well suited to computer implementations. Informally, a cut in a graph is a set of edges which, when removed from the graph, separates it into different connected components. Given a set of vertices or subgraphs called markers, the goal of these operators is to find a cut for which each induced component contains exactly one marker, and which best matches a criterion based on the image contents. For example, the criterion is often designed in such a way that the cut is located along the contours of the objects present in the image. To this aim, edges of the pixel adjacency graph can be weighted for example with the inverse of the gradient modulus. The principle of min-cut segmentation is then to find a cut (relative to the markers) which sum of edge weights is minimal [6].

The watershed is a well-known notion from the field of topography, introduced for image segmentation purposes by S. Beucher and C. Lantuéjoul [5]. Intuitively, the watershed of a function (seen as a topographical surface) is composed by the locations from which a drop of water could flow down towards different minima. In a framework of edge-weighted graphs, the watershed is defined in [9, 10] as a cut relative to the regional minima of the weight function, and which satisfies this "drop of water" principle. In [15], Meyer shows the link between minimum spanning forests and flooding algorithms, which are most often used to compute watersheds. There is indeed an equivalence between watersheds defined as cuts satisfying the drop of water principle and cuts induced by minimum spanning forests (MinSF) relative to the minima, as proved in [9, 10].

Another point of view on the watershed is studied in [13, 14]. Let us define the "length" of a path as the maximum weight of the edges along this path, then the watershed is defined by these authors as a cut which separates the components of the graph induced by a shortest-path forest rooted in the minima. This definition in terms of shortest-path forest is also the basis for the so-called fuzzy connected image segmentation [2, 16].

The goal of this paper is to clarify the links between these different optimal structures used for image segmentation. To this aim, we first give a set of definitions for these different paradigms in a same unifying framework of edge-weighted graphs. Then, we show that any MinSF is a shortest-path forest, and that the converse is, in general, not true.

At last, we prove a property which links graph cuts and watersheds, through the notion of MinSF. It is well known that the MinSFs, and hence the watersheds, are invariant if an increasing transformation is applied simultaneously to all the weights. For example, if we raise all the weights to a same positive power n, a MinSF remains a MinSF. On the contrary, min-cuts may be different for different values of n. We show that, for any weighted graph, there exists a value n such that min-cuts coincide with cuts induced by maximum spanning forests relative to the markers, furthermore, this will also be true for any number greater than n.

Proofs of the theorems presented in this paper are in [1].

1. Basic notions on graphs

In this section we state basic notions on graphs before presenting the definitions of *extension* and *cut* over a graph, which will be necessary in the sequel of the paper.

We define a graph as a pair $G = (V, E)$ where V is a finite set and E is composed of unordered pairs of elements of V, precisely, E is a subset of $\{\{x, y\} \subseteq V \mid x \neq y\}$. Each element of V is called a *node* or a *vertex (of G)*, and each element of E is called an *edge (of G)*. We denote by $G_\emptyset$ the empty graph, i.e. $G_\emptyset = (\emptyset, \emptyset)$.

Let G be a graph. If $e = \{x, y\}$ is an edge of G, we say that x and y are *adjacent (for G)*. Let $\pi = \langle x_0, \dots, x_\ell \rangle$ be an ordered sequence of nodes of G, we say that π is a *path from x_0 to x_ℓ in G (or in V)* if for any $i \in [1; \ell]$, x_i is adjacent to x_{i-1}. In this case, we say that x_0 *and x_ℓ are linked for G*. We say that π is a *simple path from x_0 to x_ℓ in G (or in V)* if π is a *path from x_0 to x_ℓ* and if all nodes of π are distinct. Notice that if there exists a path from x_0 to x_ℓ in G, then there exists a simple path from x_0 to x_ℓ. We say that G *is connected* if any two vertices of G are linked for G. Notice that $G_\emptyset$ is connected.

Let $G = (V, E)$ and $G' = (V', E')$ be two graphs. If $V' \subseteq V$ and $E' \subseteq E$ then we say that G' *is a subgraph of* G and we write $G' \subseteq G$. Notice that $G_\emptyset$ is a subgraph of any graph.

In the sequel, $G = (V, E)$ will denote a graph.

We say that X *is a connected component of* G if X is a connected subgraph of G which is maximal for this property, i.e. for any connected graph X', $X \subseteq X' \subseteq G$ implies $X' = X$. Notice that $G_\emptyset$ is not a connected component of any non-empty graph, and that $G_\emptyset$ is the connected component of, and only of, $G_\emptyset$.

Let X be a subgraph of G, we denote respectively by $V(X)$ and $E(X)$ the node set and the edge set of X.

Let X and Y be two subgraphs of G. We define $(X \cup Y) = (V(X) \cup V(Y), E(X) \cup E(Y)$ and $(X \cap Y) = (V(X) \cap V(Y), E(X) \cap E(Y))$.

Let X be a subgraph of G. An edge $\{x, y\}$ of G is *adjacent to* X if $\{x, y\} \cap V(X) \neq \emptyset$ and $\{x, y\} \notin E(X)$. In this case, if $x \in V(X)$, either $y \in V(X)$ or y is adjacent to X.

If S is a subset of E, we denote by $\overline{S}$ the complementary set of S in E, that is, $\overline{S} = E \setminus S$.

Let $S \subseteq E$. The *graph induced by* S is the graph whose edge set is S and whose vertex set is made of all points which belong to an edge in S. By abuse of notation, the subgraph induced by S will also be denoted by S.

We now present the notions of *extension* and *(graph) cut* which play an important role for optimal structures in edge-weighted graphs. The notion of extension was introduced in [4] for the case of sets. In [9, 10] this notion was extended to connected graphs. The following definition presents this notion in the case of unspecified graphs.

Definition 1 (Extension, spanning extension and cut)**.** Let G be a graph and let $G_1, G_2, \ldots, G_n$ be the connected components of G. Let M and X be two subgraphs of G. For any $i \in [1; n]$, let $M_i = M \cap G_i$ and $X_i = X \cap G_i$. We say that X *is an extension of* M if, for all $i \in [1; n]$, $M_i \subseteq X_i$ and each connected component of X_i contains exactly one connected component of M_i. We say that X *is a spanning extension of* M *(over* G*)* if X is an extension of M and if $V(X) = V$. Let $C \subseteq E$, we say that C *is a (graph) cut relative to* M *(over* G*)* if $\overline{C}$ is an extension of M over G and if C is minimal for this property (i.e. considering $D \subseteq E$, $C = D$ whenever $D \subseteq C$ and $\overline{D}$ is an extension of M over G). It may be seen that, if C is a cut, then $\overline{C}$ is necessarily a spanning extension. Moreover, if X is a spanning extension of M, then there exists a unique cut C relative to X which is called the *cut induced by* X. It may be seen that C is also a cut relative to M. □

Examples of these definitions are shown in Figure 1.

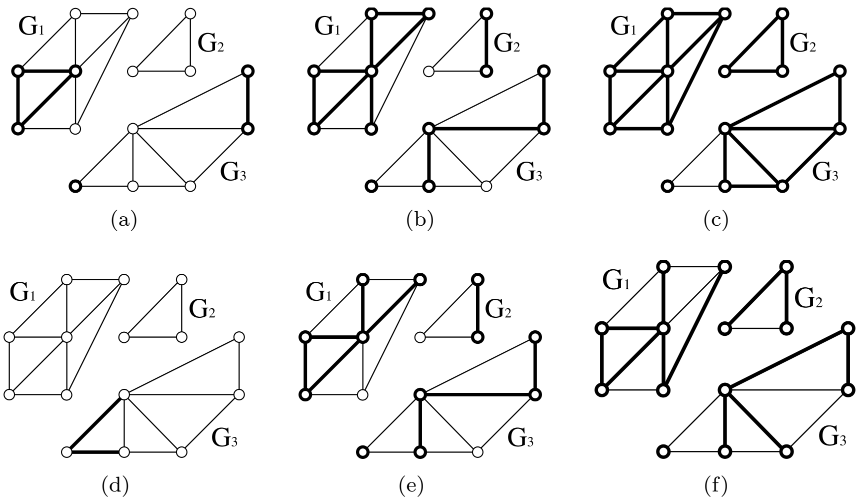


Figure 1. Graph G, composed of three connected components (G_1, G_2 and G_3), with in bold: (a) a subgraph M; (b) an extension relative to M; (c) a spanning extension relative to M; (d) a cut relative to M; (e) a forest relative to M; (f) a spanning forest relative to M.

2. Optimal structures

In this section we define the following structures: maximum spanning forests, watersheds, minimum cuts and shortest-path spanning forests.

2.1 Maximum spanning forests and watersheds

In this part, we first recall the definition of Maximum Spanning Forests (MaxSF) relative to a subgraph of G. It is shown in [10] that this notion is equivalent to the one of maximum spanning tree, which has been studied for many years in combinatorial optimization (see [8]). From this, we define the MaxSF cut and then remind the notion of watershed to highlight the link that exists between them.

Let F and M be two subgraphs of G. We say that F is a *forest relative to M* if:

- F is an extension of M, and
- for any extension $X \subseteq F$ of M, $V(X) = V(F) \Rightarrow X = F$ (i.e. we cannot eliminate an edge of F and keep the extension property).

Let F and M be two subgraphs of G. We say that F is a *spanning forest relative to M (over G)* if:

- F is a *forest relative to M*, and
- $V(F) = V$.

Equivalently, we say that F is a *spanning forest relative to M (over G)* if there exists a spanning extension X relative to M over G such that F is obtained by eliminating edges of X as long as it is possible to do it while preserving the spanning extension property.

Examples of these definitions are shown in Figures 1(e) and 1(f).

It can be seen that if G is connected and $M = (V_M, \emptyset)$ where $V_M \subseteq V$ (i.e. M is a subgraph without edge), then the notion of forest relative to M corresponds exactly to the usual notion of forest. Furthermore, if $|V(M)| = 1$ then we retrieve the usual notion of tree.

In the following, P will be a map from E to $\mathbb{R}^+$.

The pair (G, P) is an *edge-weighted graph*. If e is an edge of G, $P(e)$ is called the *altitude* or the *weight of e*. The *weight of a subgraph X of G*, denoted by $P(X)$, is the sum of its edge weights ($P(X) = \sum_{x \in E(X)} P(x)$).

Definition 2 (Maximum spanning forest)**.** Let F and M be two subgraphs of G. We say that F is a *Maximum Spanning Forest (MaxSF)* relative to M (for P) if F is a spanning forest relative to M and if the weight of F is maximum, i.e. greater than or equal to the weight of any other spanning forest relative to M. Notice that if the weight of F is minimum instead of maximum, then we have a *Minimum Spanning Forest (MinSF)*. □

Examples of this definition are shown in Figure 4.

Remark 1. Let M and F be two subgraphs of G, $f : \mathbb{R}^+ \to \mathbb{R}^+$ be a strictly increasing function and $g : \mathbb{R}^+ \to \mathbb{R}^+$ be a strictly decreasing function.
From classical results on extremal spanning forests, we know that the three following statements are equivalent:

- F is a MaxSF relative to M for P;
- F is a MaxSF relative to M for $(f \circ P)$;
- F is a MinSF relative to M for $(g \circ P)$. □

Let M be a subgraph of G and let F be a MaxSF relative to M. Since F is a spanning forest, hence a spanning extension, there exists a unique (graph) cut relative to M induced by F. We say that this cut is a *MaxSF cut relative to M*.

We now remind the definition of watersheds for a map (see [15]) and its equivalence with MinSF cuts relative to the minima of this map (see [9, 10]).

The intuitive idea underlying the notion of watershed comes from the field of topography: a drop of water falling down on a topographic surface follows a descending path and reaches a regional minimum area. The watershed may be thought of as the separating lines of the domain of attraction of drops of water.

The regions of a watershed, also called catchment basins, are associated with the regional minima of the map. In other words, each catchment basin contains a unique regional minimum, and conversely, each regional minimum is included in a unique catchment basin: the regions of the watershed are the connected components of an extension relative to the minima. They are separated by a set of edges from which a drop of water can flow down towards different minima, in the sense defined below.

A subgraph X of G is a *(regional) minimum of* P if:

- X is connected, and
- all the edges of X have the same altitude, that we will refer to as the *altitude of* X, and
- the altitude of any edge adjacent to X is strictly greater than the altitude of X.

We denote by $Min(P)$ the graph whose vertex set and edge set are, respectively, the union of the vertex sets and edge sets of all minima of P.

Let $\pi = \langle x_0, \ldots, x_\ell \rangle$ be a path in G. The path π is descending (for P) if $\forall i \in [1, \ell - 1], P(\{x_{i-1}, x_i\}) \geq P(\{x_i, x_{i+1}\})$.

Definition 3 (Watershed, Def. 3 in [9])**.** Let C be a subset of E. We say that C is a *watershed cut (for* P*)*, or simply a *watershed (for* P*)*, if $\overline{C}$ is an extension of $Min(P)$ and if for any $e = \{x_0, y_0\} \in C$, there exist $\pi_1 = \langle x_0, \ldots, x_m \rangle$ and $\pi_2 = \langle y_0, \ldots, y_n \rangle$ two descending paths in $\overline{C}$ such that:

- x_m and y_n are nodes of two distinct minima of P, and
- $P(e) \geq P(\{x_0, x_1\})$ (resp. $P(e) \geq P(\{y_0, y_1\})$), whenever $m > 0$ (resp. $n > 0$). □

Notice that a watershed is indeed a graph cut relative to $Min(P)$.

Theorem 1 (Th. 2 in [9])**.** *Let C be a subset of E. The set C is a MinSF cut relative to $Min(P)$ (for P) if and only if C is a watershed (for P).*

Any minimum spanning tree algorithm can be employed to compute a MinSF relative to a subgraph of G (see a survey in [8]). The best of them does this in quasi-linear time (see [7]), but algorithms specific to watersheds run in linear time (see [9]).

2.2 Minimum cuts (min-cuts)

In this section, we remind the notion of minimum cut.

Let M be a subgraph of G and let $C \subseteq E$. We say that C *is a minimum cut (min-cut) relative to* M *(for* P*)* if for any cut $C' \subseteq E$ relative to M, $P(C) \leq P(C')$. It can be seen that a cut C relative to M is of minimum weight if and only if $\overline{C}$ is a (spanning) extension of maximum weight relative to M. Examples of this definition are shown in Figure 4.

A fundamental result in combinatorial optimization states that, given two isolated nodes of an edge-weighted graph (called source and sink), finding a min-cut that separates these two nodes is equivalent to finding a maximum flow between them (see [12], chapter 6.2). This problem is equivalent to finding a min-cut relative to a subgraph having exactly two connected components (consider adding two extra nodes to G, the source and the sink, and highly weighted edges from each one of them to all the nodes of each of the components of M). In this case, we have polynomial-time algorithms to compute a min-cut. On the other hand, finding a min-cut relative to a subgraph with more than two connected components is NP-hard [11], but there exists approximation algorithms [6].

2.3 Shortest-path spanning forests cuts (SPSF cuts)

We now present the notion of shortest-path forest which also constitutes an optimization paradigm used for image segmentation. In particular, the image-foresting-transform [13] and the relative fuzzy-connected image segmentation [2, 17] fall in the scope of shortest-path forests. Intuitively, these methods partition the graph into connected components associated to seed points. The component of each seed consists of the points that are "more closely connected" to this seed than to any other. In many cases, in order to define the relation "is more closely connected to", we consider the length of a path π as the maximum value of an edge along π. Then, point p is more closely connected to seed s than to seed s' if the length of a shortest path from p to s is less than the length of a shortest path from p to s'. Given a set of seed points (or a seed graph), the resulting segmentation is then obtained as a shortest-path forest.

In this section, we assume that G is connected and that M is non-empty.

Let $\pi = \langle x_0, \ldots, x_\ell \rangle$ be a path in the graph G. If we have $l > 0$, we define $P(\pi) = \max\{P(\{x_{i-1}, x_i\}) \mid i \in [1;\ell]\}$. If we have $\pi = \langle x_0 \rangle$, we define $P(\pi) = \min\{P(u) \mid x_0 \in u, u \in E\}$; $P(\pi)$ is the *length of* π. Let X and Y be two subgraphs of G, we denote by $\Pi(X, Y)$ the set of all paths from X to Y in G. The *connection value between X and Y (in G and for P)*, denoted by $P(X, Y)$, is the length of a shortest path from X to Y, i.e. $P(X, Y) = \min\{P(\pi) \mid \pi \in \Pi(X, Y)\}$.

If x is a vertex of G, to simplify the notation, the graph $(\{x\}, \emptyset)$ will be also denoted by x.

Definition 4 (SPSF cut). Let M and F be two subgraphs of G. We say that F is a *shortest-path forest relative to M* if F is a forest relative to M and if, for any $x \in V(F)$, there exists, from x to M, a path π in F such that $P(\pi) = P(x, M)$. If F is a shortest-path forest relative to M and $V(F) = V$, we say that F is a *shortest-path spanning forest (SPSF) relative to M*. If F is a SPSF relative to M, the (unique) cut for F is called a *SPSF cut for M*. □

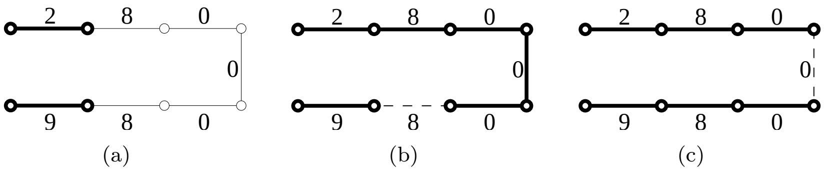


Figure 2. Graph G and map P with in bold: (a) a subgraph M; (b) a MinSF relative to M; (c) a shortest-path spanning forest relative to M which is not a MinSF relative to M.

Let G be the graph in Figure 2 and let P be the corresponding map. Let M, F and F' be the bold graphs depicted in, respectively, Figures 2(a), 2(b) and 2(c). The two graphs F and F' are SPSFs relative to M. The induced SPSF cuts for M are represented by dashed edges.

3. Some links between optimal structures

In this section, we reveal some relations existing between the different optimal structures exposed above.

3.1 Min-cut and MaxSF cut

In this section, we show that min-cuts and MaxSF cuts are linked through a modification of the map P preserving the order and emphazing the weight difference between the edges. We denote by $P^{[n]}$, and say P power n, the map from E to $\mathbb{R}^+$ defined by, for any $e \in E$, $P^{[n]}(e) = [P(e)]^n$.

Theorem 2. *If M is a subgraph of G, then there exists a real number m such that, for any $n \geq m$, any min-cut relative to M for $P^{[n]}$ is a MaxSF cut relative to M for $P^{[n]}$.*

Theorem 2 is illustrated in Figure 3 and Figure 4.

It has to be noticed that the converse of Theorem 2 is, in general, not true. See Figure 6 where the MaxSF cut relative to M for P in Figure 6(b) is not a min-cut relative to M for P, but any min-cut is a MaxSF cut. However, an intuitive interpretation of this result is to consider the MaxSF cut as a greedy heuristic to obtain a min-cut. The efficiency of this heuristic becomes higher when differences between the weights increase.

From Remark 1, we know that the MaxSF cut relative to M for $P^{[n]}$ is also a MaxSF cut relative to M for P and conversely since the change of map preserves the order.

Since we know, from Remark 1 and Theorem 1, that the watersheds are particular cases of MaxSF cuts, we deduce from Theorem 2 that the watersheds are also particular cases of the min-cuts.

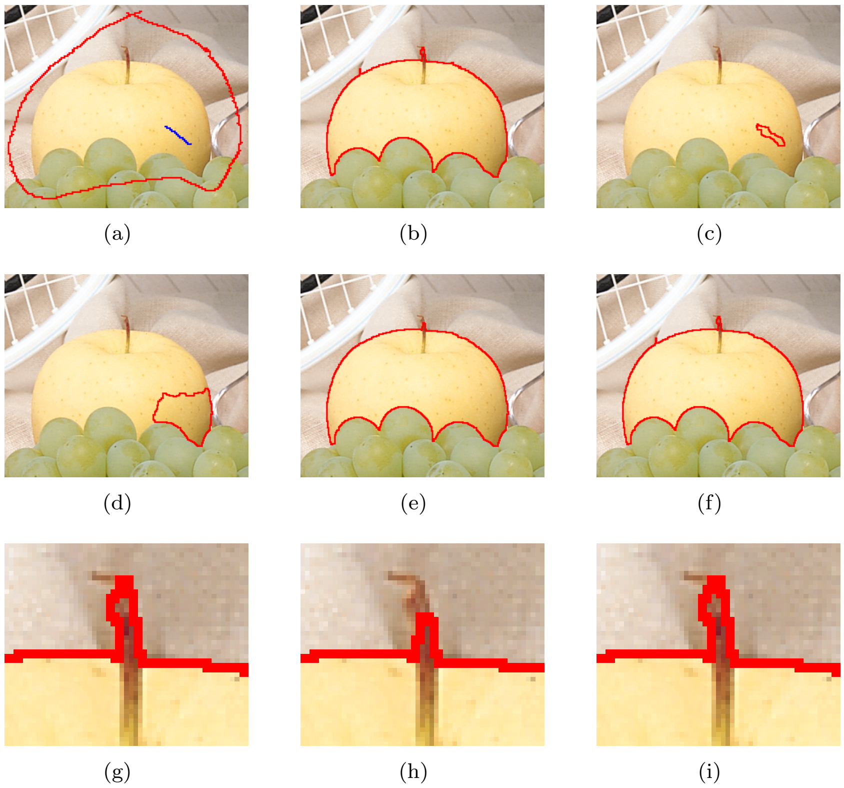


Figure 3. Color image segmentation using: (a) markers superimposed to the original image; (b) watershed on P; (c) min-cut on P; (d) min-cut on $P^{[1.4]}$; (e) min-cut on $P^{[2]}$; (f) min-cut on $P^{[3]}$; (g) zoom of watershed on P; (h) zoom of min-cut on $P^{[2]}$; (i) zoom of min-cut on $P^{[3]}$.

Figure 3 illustrates the link between these two well known segmentation paradigms through the evolution of the min-cut with different values of n. Notice that the power of the map P could then be considered as a smoothing term for the min-cut method. Indeed, when this power decreases, shortest cuts are found whereas, when it increases, longer cuts are found. These longer cuts can surround more details as well as noise. Therefore, releasing this smoothing term is not always suitable. See for example Figure 5 where the min-cut result is better than the watershed.

3.2 MinSF cuts and SPSF cuts

We now investigate the links between SPSF cuts and MinSF cuts. We show that any MinSF cut relative to a subgraph of G is a SPSF cut relative to this subgraph. Therefore, according to Theorem 2, there exist some particular

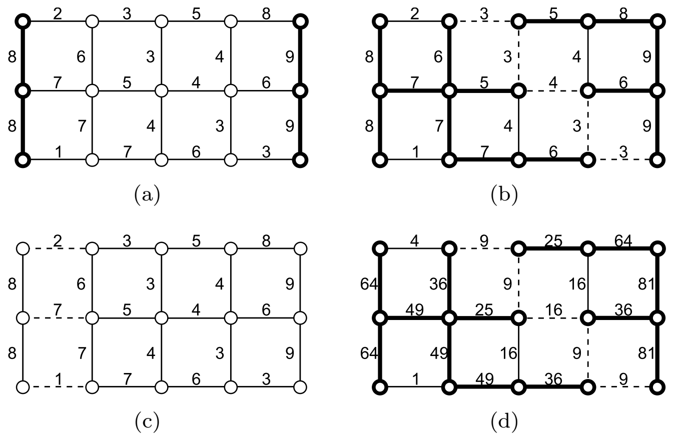


Figure 4. Graph G and map P with: (a) in bold, a subgraph M; (b) in bold, the MaxSF relative to M for P and, in dashed edges, its induced cut which, according to Remark 1 and Theorem 1, is a watershed (up to a strictly decreasing function on P); (c) in dashed edges, the min-cut relative to M for P; (d) in bold, the MaxSF relative to M for $P^{[2]}$ and, in dashed edges, its induced cut which is also the min-cut relative to M for $P^{[2]}$.

functions for which any min-cut is a SPSF cut (up to a strictly decreasing function over P). Furthermore, we prove that MinSF cuts and SPSF cuts are equivalent whenever we consider the subgraph of G which corresponds precisely to the minima of P. Hence, according to Theorem 1, this last result establishes the equivalence between the watersheds for P and the SPSF cuts relative to the minima of P.

In this section, we assume that G is connected and that M is non-empty.

Theorem 3 (Prop. 30 in [10]). *Let M and F be two subgraphs of G. If F is a MinSF relative to M, then F is a shortest-path forest relative to M. Furthermore, any MinSF cut relative to M is a SPSF cut relative to M.* [1]

The converse of the previous theorem is, in general, not true. For instance, the graph Z (Figure 2(c)), is a SPSF relative to the graph X (Figure 2(a)) whereas it is not a MinSF relative to this graph.

In fact, as stated by the following theorem, if the graph M constitutes precisely the minima of P, the equivalence between both concepts can be established.

Theorem 4 (Prop. 31 in [10]). *Let F be a subgraph of G. The graph F is a SPSF relative to $Min(P)$ if and only if F is a MinSF relative to $Min(P)$.*

[1]This result was obtained independently in [3].

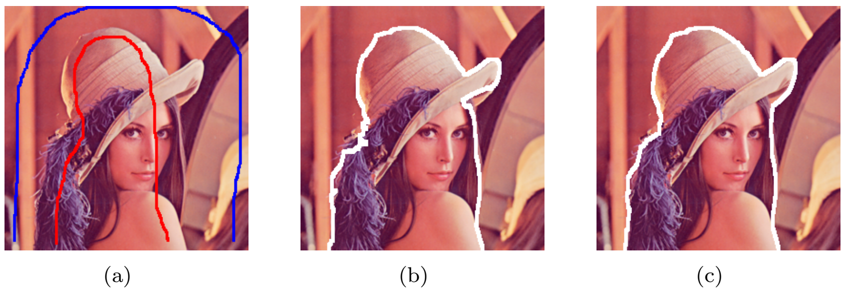


Figure 5. Color image segmentation: (a) markers superimposed to the original image; (b) watershed on P; (c) min-cut on P.

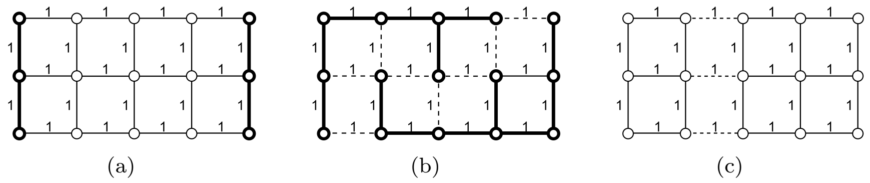


Figure 6. Graph G and map P with: (a) in bold, a subgraph M; (b) in bold, a MaxSF relative to M for P and, in dashed edges, its induced cut, which is not a min-cut; (c) in dashed edges, a min-cut relative to M for P.

Furthermore, a cut S relative to $Min(P)$ is a SPSF cut relative to $Min(P)$ if and only if S is a MinSF cut relative to $Min(P)$.

Conclusion

We compared three different optimal structures, namely extremal spanning forests, min-cuts and shortest-path forests, which have been used as the basis for popular image segmentation methods. The watershed approach, which is strongly linked to minimum spanning forests and to shortest-path forests, is also considered in this study. Although different in general, we exhibited some particular cases where a strong relation exists between these structures.

References

[1] C. Allène, J.-Y. Audibert, M. Couprie, J. Cousty, and R. Keriven, *Some links between min-cuts, optimal spanning forests and watersheds*, Technical Report IGM2007-06, Université Paris-Est, France, 2007.

[2] R. Audigier and R. A. Lotufo, *Duality between the Watershed by Image Foresting Transform and the Fuzzy Connectedness Segmentation Approaches*, SIBGRAPI, 2006, pp. 53–60.

[3] ———, *Watershed by image foresting transform, tie-zone, and theoretical relationships with other watershed definitions*, Procs. 8th International Symposium on Mathematical Morphology, 2007, pp. these proceedings.

[4] G. Bertrand, *On topological watersheds*, Journal of Mathematical Imaging and Vision **22** (2005), no. 2-3, 217–230.

[5] S. Beucher and C. Lantuéjoul, *Use of watersheds in contour detection*, Procs. of the International Workshop on Image Processing Real-Time Edge and Motion Detection/Estimation, 1979.

[6] Y. Boykov, O. Veksler, and R. Zabih, *Fast Approximate Energy Minimization via Graph Cuts*, IEEE Trans. Pattern Analysis and Machine Intelligence **23** (2001), 1222-1239.

[7] B. Chazelle, *A Minimum Spanning Tree Algorithm with Inverse-Ackermann Type Complexity*, Journal of the ACM **47** (2000), 1028–1047.

[8] T. H. Cormen, C. Leiserson, and R. Rivest, *Introduction to algorithms, second edition*, MIT Press, 2001.

[9] J. Cousty, G. Bertrand, L. Najman, and M. Couprie, *Watershed cuts*, Procs. 8th International Symposium on Mathematical Morphology, 2007, pp. these proceedings.

[10] ———, *Watershed, minimum spanning forests, and the drop of water principle*, Technical Report IGM2007-01, Université Paris-Est, France, 2007.

[11] E. Dahlhaus, D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, and M. Yannakakis, *Complexity of multiway cuts*, Procs. of the 24th Annual ACM Symposium on the Theory of Computing, 1992, pp. 241–251.

[12] R. Diestel, *Graph Theory*, Graduate Texts in Mathematics, Springer, 1997.

[13] A. X. Falcão, J. Stolfi, and R. A. Lotufo, *The image foresting transform: theory, algorithm and applications*, IEEE Trans. Pattern Analysis and Machine Intelligence **26** (2004), 19–29.

[14] R. Lotufo and A. X. Falcão, *The Ordered Queue and the Optimality of the Watershed Approaches*, Procs. of the 5th International Symposium on Mathematical Morphology, 2000, pp. 341–350.

[15] F. Meyer, *Minimum Spanning Forests for Morphological Segmentation*, Procs. of the second international conference on Mathematical Morphology and its Applications to Image Processing, 1994, pp. 77–84.

[16] P. K. Saha and J. K. Udupa, *Relative Fuzzy Connectedness among Multiple Objects: Theory, Algorithms, and Applications in Image Segmentation*, Computer Vision and Image Understanding **82** (2001), 42–56.

[17] J. K. Udupa, P. K. Saha, and R. A. Lotufo, *Relative Fuzzy Connectedness and Object Definition: Theory, Algorithms, and Applications in Image Segmentation*, IEEE Trans. Pattern Analysis and Machine Intelligence **24** (2002), no. 11, 1485–1500.

Stochastic watershed segmentation

Jesús Angulo and Dominique Jeulin

Centre de Morphologie Mathématique (CMM), Ecole des Mines de Paris, Fontainebleau, France
{jesus.angulo,dominique.jeulin}@ensmp.fr

Abstract This paper introduces a watershed-based stochastic segmentation methodology. The approach is based on using M realizations of N random markers to build a probability density function (pdf) of contours which is then segmented by volumic watershed for defining the R most significant regions. It improves the standard watershed algorithms when the aim is to segment complex images into a few regions. Three variants of the random germs framework are discussed, according to the algorithm used to build the pdf: 1) uniform random germs on the same gradient, 2) regionalised random germs on the same gradient, and 3) uniform random germs on levelled-based gradient. The last algorithm is more complex but it yields the best results.

Keywords: watershed transform, leveling, Poisson points, density of contours, random germs segmentation.

1. Introduction

Watershed transformation is one of the most powerful tools for image segmentation. Starting from a gradient, the classical paradigm of watershed segmentation consists in determining markers for each region of interest. The markers avoid the over-segmentation (a region is associated to each minimum of the function) and moreover, the watershed is relatively robust to marker position [2]. The markers-based watershed is appropriate for interactive segmentation. Several watershed-based hierarchical approaches allow addressing fields where the markers cannot be easily defined (e.g., multimedia applications). Mainly, two hierarchical techniques can be distinguished: 1) non-parametric waterfalls algorithm [3] and 2) hierarchies based on extinction values, which allows to select the minima used in the watershed according to morphological criteria (dynamics, surface area and volume) [10, 15].

The volume-based hierarchical segmentation is particularly useful in many applications aiming at segmenting natural images since the volume, which combines the criteria of dynamics and area, selects the most significant regions from a visual viewpoint. However, the performance of the approach decreases drastically when the image is segmented in very few regions, which is just the goal of several applications (e.g., segmentation-based image indexing). Figure 1 gives four colour images segmented by volumic watershed into $R = 10$, 20 and 50 regions and we can observe that many important regions are not well determined (even when $R = 50$). The classical solution involves to filter out the image in order to simplify the details and to enhance the main regions (typically using morphological filters such as levelings [11]).

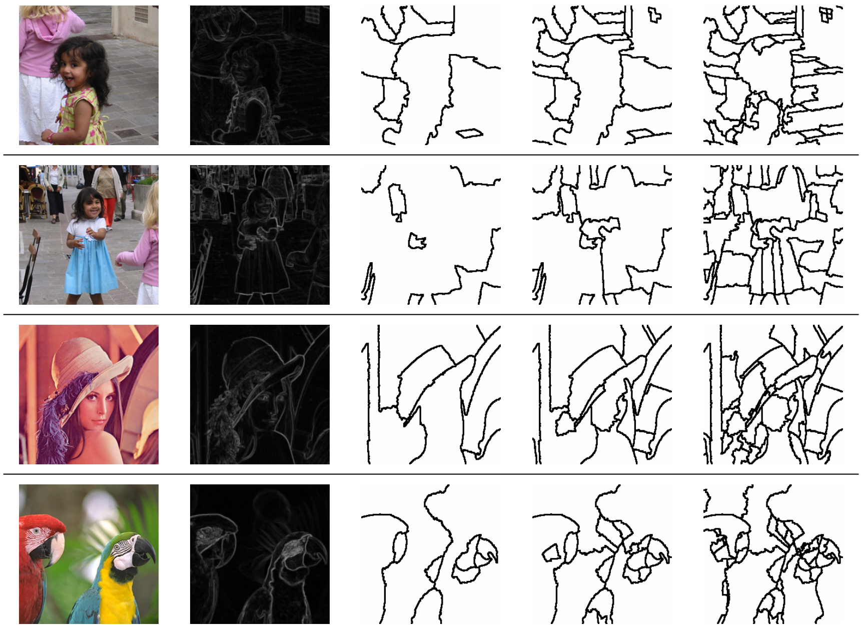


Figure 1. Examples of colour images segmented by means of volumic watershed into the R most significant regions. First column, original images $\mathbf{f}$; second column, colour gradient $\varrho^{LS+H}(\mathbf{f})$; third column, $sg^{R-vol}(\varrho^{LS+H}(\mathbf{f}), 10)$; fourth column $sg^{R-vol}(\varrho^{LS+H}(\mathbf{f}), 20)$, and last column $sg^{R-vol}(\varrho^{LS+H}(\mathbf{f}), 50)$.

In fact, the problem lies in the deterministic criterion of volume, computed for each minimum of the function to flood, which depends on the local image information; and nevertheless, the final flooding watershed is a competition between the different minima to determine the optimal partition (in fact, it is the solution of shortest path problem when the path cost is given by the maximum of the arc weights in the path [10]). The aim of this paper is to introduce a watershed-based probabilistic framework to detect the contours which are robust with respect to variations in the segmentation conditions. More precisely, we explore here a stochastic approach based on using random markers to build a probability density function of contours which is then segmented by volumic watershed for defining the most significant regions. Keeping in mind that the goal is the unsupervised segmentation of natural images in very few regions.

The probabilistic segmentation has been already studied in the literature, for instance using cooccurrence probability on graphs [5], Bayesian framework [16], Markov Random Fields [7] (combined with watershed segmentation [8]), Markov Chain Monte Carlo [14]. But to our knowledge, this is first study of probabilistic segmentation based on random markers simulations for watershed transformation. The closest previous work to our study is [13], where the sum of watersheds from a series of polarimetric images was used to define a final distribution of contours.

2. Basic notions and operators

Watershed segmentation. The function used in the watershed transformation is the image gradient. In this paper, the aim is to segment colour images and hence, according to our previous works [1], we propose to compute a colour gradient in a luminance/saturation/hue (LSH) representation, presenting better performances that other colour gradients. But any other colour gradient can be also used, including for instance marginal gradients in RGB. Let $\mathbf{f}(x) = (f_L(x), f_S(x), f_H(x))$ be a colour image in the LSH representation, the colour gradient is given by $\varrho^{LS+H}(\mathbf{f}(\mathbf{x})) = f_S(x) \times \varrho^{\circ}(f_H(x)) + (1 - f_S(x)) \times \varrho(f_L(x)) + \varrho(f_S(x))$, where $\varrho(g(x))$ is the morphological gradient of the scalar function $g(x)$ and $\varrho^{\circ}(a(x))$ is the circular centred gradient of the angular function $a(x)$.

Two watershed algorithms are used in this study. Let $mrk(x)$ be the image of markers, the binary image of segmentation contours associated to these markers, and according the colour gradient $\varrho^{LS+H}(\mathbf{f}(x))$, is denoted by $sg^{mrk}(\varrho^{LS+H}(x))$. Using the same gradient, the volumic-based segmentation into R regions is named $sg^{R-vol}(\varrho^{LS+H}(x))$.

Leveling. The leveling $\lambda(mrk, f)$ of a reference function f and a marker function mrk ($f(x)$ and $mrk(x)$ are two grey level images) can be computed by means of an iterative algorithm with geodesic dilations/erosions [11]. Several extensions to colour images have been proposed for levelings. We propose for this study to apply a marginal approach in RGB, which consists in computing a separated leveling for each red/green/blue colour component, i.e. the colour leveling of image $\mathbf{f}(x) = (f_R(x), f_G(x), f_B(x))$ according to the markers $mrk(x)$ is the colour image $\lambda(\mathbf{f}, mrk) = (\lambda(mrk, f_R), \lambda(mrk, f_G), \lambda(mrk, f_G))$. The marginal approach introduces false colour but this is not critical for segmentation purposes.

Generation of random germs. The paradigm of watershed segmentation lays on the appropriate choice of markers, which are the seeds to generate basins of attraction [2,3]. It is claimed, and known from practice, that the most intelligent part of this technique of segmentation resides in the development of criteria used to select the required markers. In the present approach, we follow an opposite direction, by selecting random germs for markers. This arbitrary choice will be balanced by the use of a given number M of realizations, in order to filter out non significant fluctuations.

A rather natural way to introduce random germs [9] is to generate realizations of a Poisson point process with a constant intensity (namely average number of points per unit area) θ. It is well known that the random number of points N falling in a domain D with area $|D|$ follows a Poisson distribution with parameter $\theta\,|D|$. In addition, conditionally to the fact that $N = n$, the n points are independently and uniformly distributed over D. In what follows, we will fix the value N of the number of random germs (instead of using a random number as for the Poisson point process), and we will generate independent realizations of the location of the germs in D. In some cases, as will be illustrated below, it may be interesting to generate a non-uniform distribution of germs, with a regionalised intensity (or measure) $\theta(x)$. In the Poisson case, N follows a Poisson distribution with parameter $\theta(D)$, and conditionally to the fact that $N = n$, the n points are independently distributed over D with the probability density function $\theta(x)/\theta(D)$. In what follows, the intensity $\theta(x)$ will be generated from the image, and we will use a fixed number of germs N, as for the homogeneous case.

Parzen method to calculate a pdf. The kernel density estimation, or Parzen window method [6], is a way of estimating the probability density function (pdf) of a random variable. Let $\mathbf{x}_1$,$\mathbf{x}_2$, $\cdots$, $\mathbf{x}_M \in \mathbb{R}^n$ be M samples of a random variable, the kernel density approximation of its pdf is: $\hat{f}_h(\mathbf{x}) = \frac{1}{Nh}\sum_{i=1}^{N} K(\frac{\mathbf{x}-\mathbf{x}_i}{h})$, where $K(\mathbf{x})$ is some kernel and the bandwidth h a smoothing parameter. Usually, $K(x)$ is taken to be a Gaussian function with mean zero and variance σ^2, which determines the smoothing effect.

3. Uniform random germs segmentation

Let $\{mrk_i(x)\}_{i=1}^{M}$ be a series of M realizations of N uniform random markers. Each one of these binary images of points is considered as the markers for a watershed segmentation of colour gradient $sg^{mrk}(\varrho^{LS+H}(x))$ and consequently, a series of segmentations is obtained, i.e., $\{sg_i^{mrk}(x)\}_{i=1}^{M}$, see Figure 2. Note that the number of points determines the number of regions obtained (i.e., essential property of watershed transformation). As we can observe from the example, the main contours appear regardless of the position of germs.

Starting from the M realizations of contours, the probability density function of contours is computed by Parzen window method. The smoothing effect of the Gaussian kernel (typically $\sigma = 3$) is important to obtain a function where closed contours (e.g., in textured regions or associated to small regions) are added together. The $pdf(x)$ could be thresholded in order to obtain the most prominent contours, however the result are only pieces of contours (not enclosing regions). In addition, we have studied the histograms for several examples and there is not an optimal threshold to separate the classes of contours.

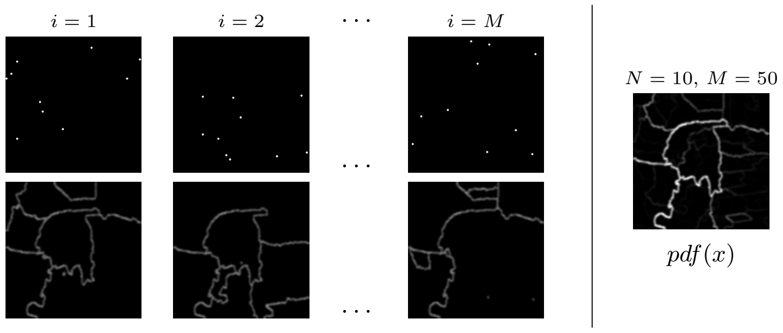


Figure 2. Left, M realizations of N uniform random germs, $mrk_i(x)$, and corresponding marker-based watershed contours, $sg_i^{mrk}(x)$. Right, probability density function of contours computed by Parzen window method for $N = 10$ and $M = 50$.

The main drawback of using a uniform distribution of the random markers is to induce an over-segmentation of the largest watersheds, since the average number of germs falling in a given region is proportional to its area. This is avoided by means of a volume-watershed segmentation, or by using a regionalised intensity of germs, as illustrated later. We propose to partition the $pdf(x)$ of contours with the volume-based watershed to obtain the R most significant regions, i.e., $sg^{R-vol}(pdf(x), R)$. Each catchment basin (each minima) of $pdf(x)$ corresponds to one the regions of the sum (or union) of the different $sg_i^{mrk}(x)$ and the integral of each catchment basin corresponds to the probability to be region of the segmentation. Consequently, the volumic watershed of $pdf(x)$ yields the regions according to their probabilities. In Figure 3 is given a comparison of segmentation into $R = 10$, 20 and 50 regions for two different $pdf(x)$. The results should be compared with those associated to $sg^{R-vol}(\varrho^{LS+H}(x), R)$ (see Figure 1). A property of the Gaussian filter, observed from the examples, is the regularisation of $pdf(x)$ which involves relatively rounded watershed contours.

3.1 Influence of parameters N and M

From the examples of Figure 1 and other similar results, we state that the method hardly depends on the number of realizations M, which is a good characteristic to guarantee its robustness. In practice, we have verified that the $pdf(x)$ converges to a stable distribution of contours even for low values of M (20 or 50). We propose in any case, to take a higher value, typically $M = 100$ or 200 in order to obtain more regular contours.

The random points explore uniformly the image space and the choice of N is important to fix the degree of stochastic sampling (note that the probability depends on the ratio between N and the image size or number of pixels). Moreover, if the value of N is low, a segmentation into large regions

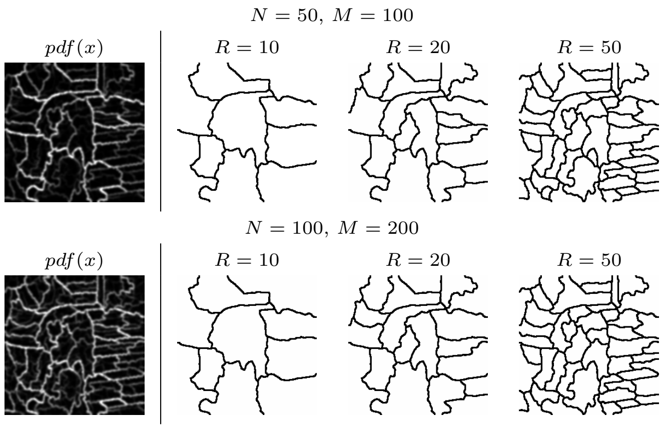


Figure 3. Left, probability density function of contours, $pdf(x)$ for N unifom germs and M realizations. Right, volumic watershed-based segmentation of $pdf(x)$ into the R most significant regions, $sg^{R-vol}(pdf), R)$.

is privileged; instead of a high value of N will produce smaller regions. If N is too high, the over-segmentation of sg_i^{mrk} leads to a very smooth $pdf(x)$, which loss its property for selecting the R contours. In fact, we can conclude that the uniform germs segmentation is mainly depending on parameter N which is related to R (number of regions to be determined) and it is logical to take $N > R$. But again, the method is quite robust to the choice of N: from the examples of images of size 256×256 to be segmented into $R = 10$, 20 or 50 the choice of $N = 50$ or 100 produces exactly the same results.

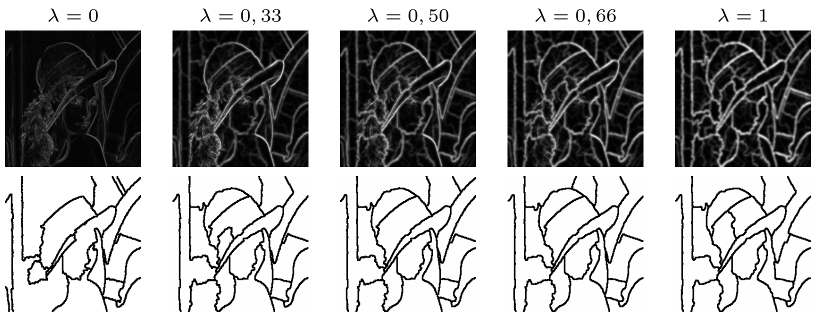


Figure 4. First row, probabilistic gradient $\rho(x)$ (i.e., linear combination of colour gradient and pdf) for different values of λ and second row, associated volumic watershed-based segmentation into $R = 20$ regions.

3.2 Probabilistic gradient

The function $pdf(x)$ can be combined with the initial gradient in order to reinforce the gradient contours which have a high probability: $\rho(x) = \omega_1 \varrho^{LS+H}(\mathbf{f}(x)) + \omega_2 pdf(x)$, considering a typical barycentric combination (both functions defined in $[0,1]$), i.e., $\omega_1 = (1-\lambda)$ and $\omega_2 = \lambda$.

We have studied the behaviour of $\rho(x)$ for volumic segmentation, i.e., $sg^{R-vol}(\rho(x))$, with respect to the value of control λ (note that for $\lambda = 0$ the gradient is obtained and for $\lambda = 1$, exclusively the probability density function of contours). In Figure 4 is shown an example of segmentation into 20 regions for different λ. It is observed that, even for low values of λ, the results of segmentation are notably improved. This is coherent with the fact that the $pdf(x)$, derived from the gradient, contains all the useful information for the segmentation. In any case, we have confirmed on the basis of many other examples that when $\lambda = 0,5$ (averaged combination) the results are in general more satisfactory.

4. Regionalised random germs segmentation

In the previous algorithm, the random germs are uniformly distributed in the image domain. We have also studied how a regionalised distribution of germs could be used to build the distribution of contours. The first question to deal with is the choice of the regionalisation function $\theta(x)$.

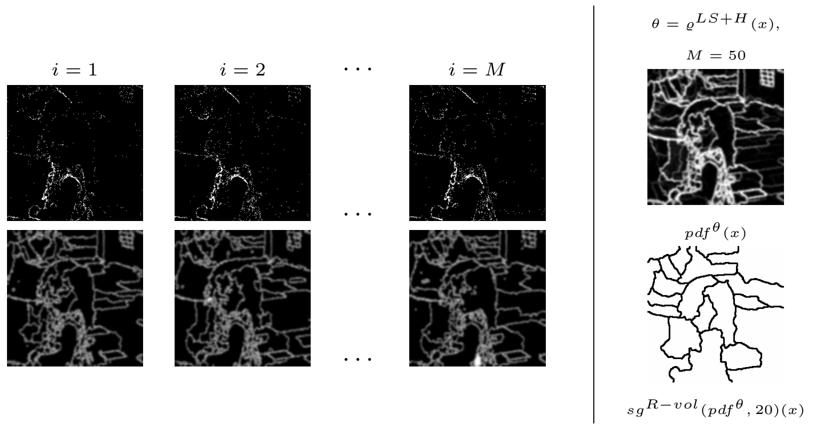

$i = 1$
$i = 2$
$\cdots$
$i = M$
$\theta = \varrho^{LS+H}(x)$,
$M = 50$
$pdf^{\theta}(x)$
$sg^{R-vol}(pdf^{\theta}, 20)(x)$

Figure 5. Left, M realizations of regionalised random germs (the function of regionalisation is the colour gradient $\theta = \varrho^{LS+H}$), $mrk_i(x)$, and corresponding marker-based watershed contours, $sg_i^{mrk^{\theta}}(x)$. Right, probability density function of contours computed by Parzen window method for $M = 50$, and segmentation of $pdf^{\theta}(x)$ into $R = 20$ volumic regions.

Several alternatives are possible. We can for instance use the component of luminance $\theta(x) = f_L(x)$ (respectively, the negative of luminance $\theta(x) = f_L^c(x)$), in such a way that the bright regions (respectively, the dark regions) will produce random germs. It is evident that this kind of regionalisation is not very useful for segmentation. It seems more natural to work on the colour gradient, $\theta(x) = \varrho^{LS+H}(x)$. In this case, the germs of $mrk_i^{\theta}(x)$ are located around the zones of high gradient value, that is the zones closed to the contours. Once the series of M contours $sg_i^{mrk^{\theta}}(x)$ is computed, the corresponding probability density of contours $pdf^{\theta}(x)$ is obtained by the Parzen window method. As previously, this function is finally segmented by volume-based watershed, see the example of Figure 5. The regionalised segmentation depends on the properties of dynamics of colour gradient. Moreover, the different random point realizations using the same $\theta(x)$ are quite similar and consequently, the realizations of contours too. By this regionalised sampling, another characteristic of the obtained $pdf^{\theta}(x)$ is that the distribution is very similar to the gradient, but where all the contours are enhanced. The final results of segmentation for $sg^{R-vol}(pdf^{\theta}, R)$ are in any case better than for $sg^{R-vol}(\varrho^{LS+H}, R)$. We have also evaluated the interest of $\theta(x)$ equal to the negative of the gradient (i.e., locating germs in low gradient zones); however in this case too many germs are introduced in each realization and the over-segmentation involves useless pdf's.

5. Uniform random germs leveling and segmentation

The morphological connected filters suppress details but preserve the contours of the remaining objects. Levelings are a subclass of symmetric connected operators which are very useful to simplify an image before segmentation by watershed transformation [11]. In fact, the image marker for the leveling is a rough simplification of original image. Pushing our approach to the limit, the rationale behind the last variant of the proposed stochastic segmentation is based on using the random germs as markers before for the leveling, in order to obtain a very simplified gradient on which is computed the watershed with the same markers.

The steps of this algorithm are summarised as follows (see Figure 6).

- To throw the M realizations of N uniform random germs: $\{mrk_i(x)\}_{i=1}^{M}$.
- To compute the leveling for the colour image associated to each image of germs: $\mathbf{lev}_i(x) = \lambda(\mathbf{f}, mrk_i)$.
- To calculate the series of colour gradients associated to the leveled colour image: $\varrho_i(x) = \varrho_i^{LS+H}(\mathbf{lev}_i)$.
- Each colour gradient ϱ_i is segmented with the markers mrk_i: $sg_i^{lev-mrk}(x) = sg^{mrk_i}(\varrho_i)$.

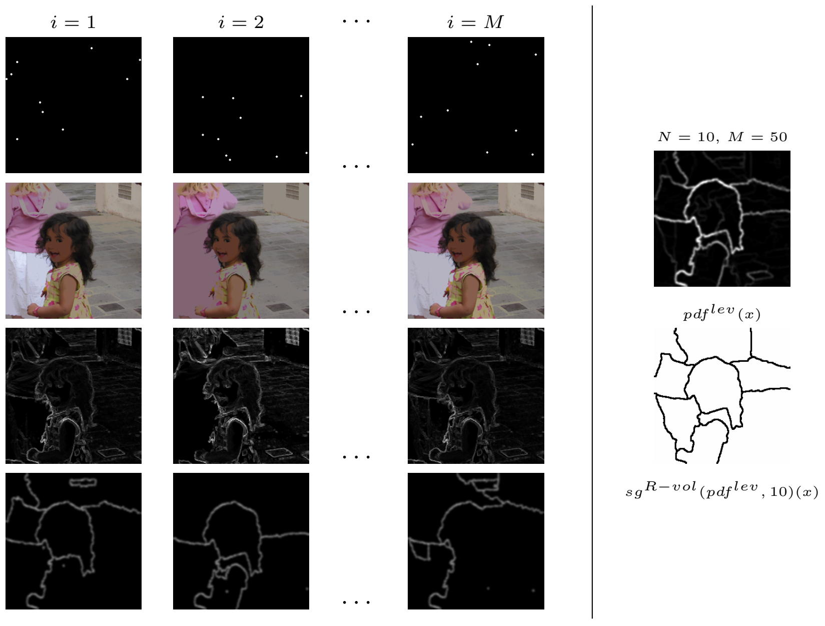


Figure 6. Left, M realizations of N uniform random germs, $mrk_i(x)$, marginal colour levelings using the random germs as markers, $lev_i(x)$, associated colour gradients, $\varrho_i(x)$ and corresponding marker-based watershed contours, $sg_i^{lev-mrk}(x)$. Right, probability density function of contours computed by Parzen window method for $N = 10$ and $M = 50$, and segmentation of $pdf^{lev}(x)$ into $R = 10$ volumic regions.

- To obtain the probability density function of contours: $pdf^{lev}(x) = \frac{1}{M}\sum_{i=1}^{M} sg_i^{lev-mrk} * G_\sigma$.
- Let $\widehat{\varrho}(x) = \frac{1}{M}\sum_{i=1}^{M} \varrho_i(x)$ be the averaged colour gradient for the M realizations, to compute the leveling-based probabilistic gradient which is defined as follows: $\rho^{lev}(x) = (1-\lambda)\widehat{\varrho}(x) + \lambda pdf^{lev}(x)$ (typ. $\lambda = 0.5$).
- To segment by volumic watershed into R regions the function of contours $sg^{R-vol}(pdf(x))$ (or the probabilistic gradient $sg^{R-vol}(\rho^{lev}(x))$).

Figure 7 shows a final comparison with examples of colour images segmented by volumic watershed ($R = 10$, 20 and 50) on probabilistic gradient $\rho(x)$ and on leveling-based probabilistic gradient $\rho^{lev}(x)$ (both with $\lambda = 0.5$). These results should be compared with those of Figure 1. It is evident that the stochastic algorithms proposed in this study yields to better segmentations than the standard watershed. It is observed also that the last method including levelings results in very good image partitions.

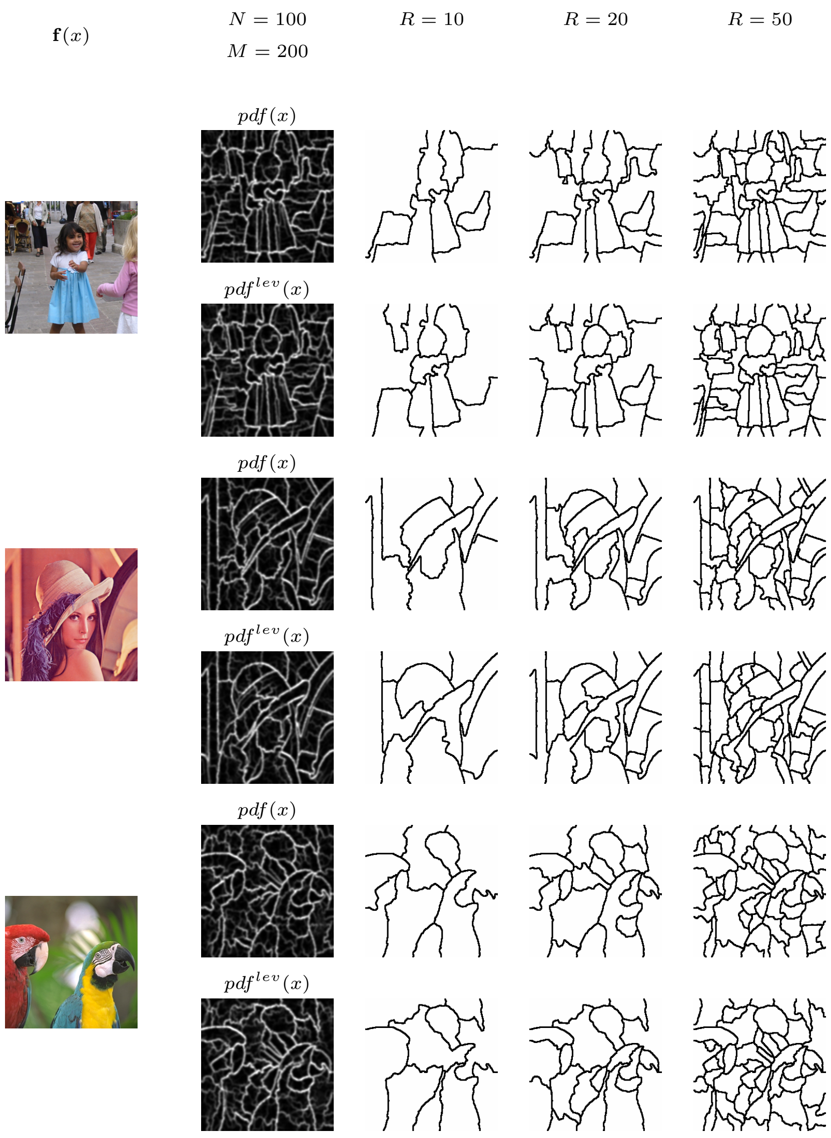


Figure 7. Examples of colour images segmented by means of volumic watershed into the R most significant regions ($R = 10$, 20 and 50) on probabilistic gradients $\rho(x)$ and $\rho^{lev}(x)$ (both with $\lambda = 0.5$) and derived from $pdf(x)$ and $pdf^{lev}(x)$ respectively (both for $N = 100$ and $M = 200$).

6. Implementation issues

The M realizations of uniform/regionalised random germs contours are obtained from the same function (i.e., colour gradient) using different markers. Consequently, working on the neighbourhood graph of catchment basins and its minimum spanning tree (MST) [12], the N random markers can be considered as N random nodes of the MST instead of N image points. Two main advantages are associated to the graph implementation: firstly, a fast computation of M segmentations from different markers on the same MST; and secondly, the control of watershed bias which could be associated to the random positions of markers [4].

The algorithm using the uniform random germs as markers, first for the levelling and then for the watershed has an upper computational load (time of computation). Moreover, in each realization, the gradient is different (i.e., a different graph) and therefore the MST cannot be reused. In any case, nowadays using the fast implementations of watershed algorithms (100 ms for a 256×256 images running on a current standard Laptop), the time of execution to segment a colour image according our stochastic framework is around 10 s.

7. Discussion and conclusions

We have introduced in this paper a new morphological stochastic segmentation approach which improves the standard watershed algorithms when the aim is to segment complex images into a few regions. The improvement in the segmentation is less important for images presenting specific objects on a homogenous background. We have illustrated three variants of the random germs framework, according to the algorithm used to build the probability density of contours: uniform random germs on the same gradient, regionalised random germs on the same gradient and uniform random germs on levelled-based gradient. The last algorithm is more complex but it yields the best results.

In ongoing research, we consider to explore other variants using evolved random point simulations (structural grids, conditional models, etc.), working on a multi-scale framework (image pyramids and image decompositions). We are also working on probabilistic approaches combining colour gradients and texture information. In the last case, probabilistic rules of aggregation in the construction of the watersheds from the random seeds would introduce a second level of randomness in the segmentation process.

From a fundamental viewpoint, the idea behind our approach is that there are two types of contours associated to the watershed of a gradient: *1st order contours*, which correspond to "significant" regions and which are relatively independent from markers; and *2nd order contours*, associated to "small", "low contrasted" or "textured" regions and which depend strongly on the place of markers. Our probabilistic framework aims at enhancing the 1st order contours from a sampling effect, to improve the result of watershed. It should be interesting to study if it is possible to determine by deterministic methods the type of each contour present in an image.

References

[1] J. Angulo and J. Serra, *Modelling and Segmentation of Colour Images in Polar Representations*, Image Vision and Computing **25** (2007), no. 4, 475–495.

[2] S. Beucher and F. Meyer, *The Morphological Approach to Segmentation: The Watershed Transformation*, New York (1992), Mathematical Morphology in Image Processing, (E. Dougherty Ed.), pp. 433–481.

[3] S. Beucher, *Watershed, hierarchical segmentation and waterfall algorithm*, ISMM'94 (1994), Mathematical Morphology and its Applications to Image and Signal Processing, pp. 69–76.

[4] S. Beucher, *Algorithmes sans biais de Ligne de Partage des Eaux*, Ecole des Mines de Paris, Internal Note CMM, 34 p., Februry 2002.

[5] K. Cho and P. Meer, *Image Segmentation from Consensus Information*, Computer Vision and Image Understanding **68** (1996), no. 1, 72–89.

[6] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, Wiley, New York, 1973.

[7] S. Geman and D. Geman, *Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images*, IEEE Trans. on Pattern Analysis and Machine Intelligence **6** (1984), 721–741.

[8] V. Grau, A. U. J. Mewes, M. Alcañiz, R. Kikinis, and S. K. Warfield, *Improved Watershed Transform for Medical Image Segmentation Using Prior Information*, IEEE Trans. on Medical Imaging **23** (2004), 447–458.

[9] D. Jeulin, *Modèles morphologiques de structures aléatoires et de changement d'échelle*, Université Caen, Thèse Doctorat ès Sciences Physiques, 1991.

[10] F. Meyer, *An Overview of Morphological Segmentation*, International Journal of Pattern Recognition and Artificial Intelligence **15** (2001), no. 7, 1089–1118.

[11] ———, *Levelings, Image Simplification Filters for Segmentation*, Journal of Mathematical Imaging and Vision **20** (2004), 59–72.

[12] ———, *Grey-weighted, ultrametric and lexicographic distances*, ISMM'05 (2005), Mathematical Morphology and its Applications to Image and Signal Processing, pp. 289–298.

[13] J. Serra and M. Mlynarczuk, *Morphological merging of multidimensional data*, STERMAT'00 (2000), 6th International Conference Stereology and Image Analysis in Materials Science, pp. 385–390.

[14] Z. Tu and S.-C. Zhu, *Image Segmentation by Data-Driven Markov Chain Monte Carlo*, IEEE Trans. on Pattern Analysis and Machine Intelligence **24** (2002), 657–673.

[15] C. Vachier and F. Meyer, *Extinction value: a new measurement of persistence*, (1995), IEEE Workshop on Nonlinear Signal and Image Processing, pp. 254–257.

[16] K. L. Vincken, A. S. E. Koster, and M. A. Viergever, *Probabilistic Multiscale Image Segmentation*, IEEE Trans. on Pattern Analysis and Machine Intelligence **19** (1997), 109–120.

Watershed by image foresting transform, tie-zone, and theoretical relationships with other watershed definitions

Romaric Audigier and Roberto de A. Lotufo

Faculdade de Engenharia Elétrica e de Computação (FEEC), Universidade Estadual de Campinas (Unicamp), SP, Brasil
`audigier@dca.fee.unicamp.br, lotufo@unicamp.br`

Abstract To better understand the numerous solutions related to watershed transform (WT), this paper shows the relationships between some discrete definitions of the WT: the watersheds based on image foresting transform (IFT), topographic distance (TD), local condition (LC), and minimum spanning forest (MSF). We demonstrate that the tie-zone (TZ) concept, that unifies the multiple solutions of a given WT, when applied to the IFT-WT, includes all the solutions predicted by the other paradigms: the watershed line of TD-WT is contained in the TZ of the IFT-WT, while the catchment basins of the former contain the basins of the latter; any solution of LC-WT or MSF-WT is also solution of the IFT-WT. Furthermore, the TD-WT can be seen as the TZ transform of the LC-WT.

Keywords: image segmentation, watershed transform, graph theory, minimum spanning forest, shortest-path forest.

1. Introduction

The *watershed transform* (WT) is a famous and powerful segmentation tool in morphological image processing. First introduced by Beucher and Lantuéjoul [7] for contour detection and applied in digital image segmentation by Beucher and Meyer [8], it is inspired from a physical principle well-known in geography: if a drop of water falls on a topographic surface, it follows the greatest slope until reaching a valley. The set of points which lead to the same valley is called a *(catchment) basin. Watershed lines* separate different basins. In the WT, an image is seen as a topographic surface where gray level corresponds to altitude. In practice, the topography is made of a gradient of the image to segment. In this case, it is expected that a region with low gradient, a valley, corresponds to a rather homogeneous region and possibly to the same object. Ideally, basins correspond to segmented objects separated by watershed lines.

Many definitions and numerous algorithms for WT exist in literature. Furthermore, multiple WT solutions are sometimes returned by an algorithm according to its implementations or even by the theoretical definition itself. This disconcerting fact motivated the investigation of the relationships between theoretical WT definitions.

Definitions in continuous space have been proposed [7, 18, 19, 21] and consider the watershed as a skeleton by influence zones (SKIZ) generalized to gray-scale images. In discrete space (of interest in this paper), there are many definitions which can be classified in five main paradigms. The *WT based on local condition* (LC-WT) mimics the intuitive drop of water paradigm. The inclusion of a pixel to a basin is achieved by iteratively respecting a local condition of label continuity along a path of steepest descent that reaches the basin minimum. It is why this definition includes algorithms of “arrowing”, “rain simulation”, “downhill”, “toboggan”, “hill climbing” [14, 20, 22]. The variation among them is due to processing strategy (ordered or unordered data scanning, depth- or breadth-first, union-find) and data structure.

The *WT based on flooding* has a recursive definition [23] that simulates the immersion of a topography representing the image. At each flooding level, growing catchment basins invade flooded regions that belong to their respective influence zone. The watershed corresponds to the SKIZ.

The *topological WT* [10] cannot be viewed as a generalized SKIZ but in fact, as the ultimate homotopic thinning that transforms the image while preserving some topological properties as the number of connected components of each lower cross-section and the saliency between any two (basin) minima.

The *WT based on path-cost minimization* associates a pixel to a catchment basin when the topographic distance is strictly minimum to the respective regional minimum in the case of the *WT by topographic distance* (TD-WT) [18]; or it builds a forest of minimum-path trees, each tree representing a basin, in the case of the *WT by image foresting transform* (IFT-WT) [12, 15].

The *WT based on minimum spanning forest* (MSF-WT) associates a graph to an image and builds a MSF [17], i.e., a spanning forest minimizing the sum of the weights of the arcs used for its construction. Trees correspond to basins.

Table 1 summarizes some characteristics of these WT definitions. Only flooding-WT and TD-WT definitions (not the related algorithms) return unique solution (Figure 1(b, i)), but the concept of tie zone (TZ) can be applied to the IFT-WT to unify the set of multiple solutions by creating litigious zones when solutions differ.

The LC-WT, IFT-WT (Figure 1(e–h)) and MSF-WT, are sometimes called “region”-WT because all pixels are assigned to basins, by definition. Watershed lines are considered as located between basin pixels, but can be visualized by *ad-hoc* algorithms. The other definitions are known as “line”-WT because some pixels are labeled as watershed. Yet, except for the topological WT definition (Figure 1(c, d)), they do not define lines that consistently separate basins but, instead, possibly thick and disconnected watershed lines.

Table 1. Characteristics of the main watershed transform (WT) definitions.

Watershed definitions	Unique solution	Watershed pixels	Separating lines	Thin lines	Grayscaled lines
LC-WT	no	no	—	—	—
Flooding-WT	yes*	yes	no	no	no
Topological-WT	no	yes	yes	no	yes
TD-WT	yes*	yes	no	no	no
IFT-WT	no	no	—	—	—
TZ-IFT-WT	yes	tie-zone	no	no	no
MSF-WT	no	no	—	—	—

* The strict definitions have a unique solution but the algorithms derived in [8, 23] do not respect the definitions and, therefore, return multiple solutions.

Observe that among these paradigms, TD-WT, IFT-WT and MSF-WT are based on a global optimality criterion. Both IFT-WT and MSF-WT are only defined in discrete space. The other paradigms attempt to mimic a continuous definition, i.e., they may be defined in both discrete and continuous spaces.

This paper shows the relationships between the discrete definitions of IFT-WT, TD-WT, MSF-WT and LC-WT. We show that the TZ watershed, derived from the solutions of the IFT-WT, contains all the solutions predicted by the other paradigms.

In Section 2, we present the IFT-WT formalism, and the TZ concept. Section 3 recalls the definition of LC-WT and demonstrates that any solution of LC-WT is also solution of the IFT-WT. Section 4 shows that the watershed region of TD-WT is contained in the TZ (derived from the IFT-WT), and the basins of the former contain the basins of the latter. In addition, the TD-WT can be seen as the TZ transform of the LC-WT. Finally, Section 5 demonstrates that any solution of MSF-WT is also solution of the IFT-WT.

2. The image foresting transform (IFT)

The IFT is a general framework based on graph theory in which an image is seen as a graph and pixels (or voxels) as its nodes. This transform returns a *shortest path forest* (SPF) from an input image-graph. Depending on the path-cost function utilized and other input parameters (adjacency, arc weights), the IFT can compute different image processing operations [11, 12]: distance transforms, connected filters, interactive object delineation ("live-wire"), segmentation by fuzzy connectedness [3] and segmentation by watershed.

$\underline{0}$	$\underline{0}$	2	1	$\underline{0}$
$\underline{0}$	4	6	5	1
9	7	4	4	3
9	6	3	5	3
$\underline{0}$	8	2	7	$\underline{0}$
$\underline{0}$	$\underline{0}$	1	$\underline{0}$	$\underline{0}$

(a) Input image

A	A	**W**	B	B
A	A	**W**	B	B
A	**W**	B	B	B
W	B	B	**W**	D
C	**W**	B	**W**	D
C	C	**W**	D	D

(b) Flooding-WT

0	0	**2**	0	0
0	0	0	**2**	0
0	0	0	0	**2**
3	0	**3**	0	**3**
0	**3**	**1**	**3**	0
0	0	**1**	0	0

(c) Topological-WT 1

0	0	**2**	0	0
0	**2**	0	0	0
2	0	0	0	0
3	0	**3**	0	**3**
0	**3**	**1**	**3**	0
0	0	**1**	0	0

(d) Topological-WT 2

A	A	A	B	B
A	A	A	B	B
A	A	C	B	B
C	C	C	C	D
C	C	C	D	D
C	C	C	D	D

(e) LC-WT 1

A	A	A	B	B
A	A	A	B	B
A	D	D	B	B
C	D	D	D	D
C	C	D	D	D
C	C	D	D	D

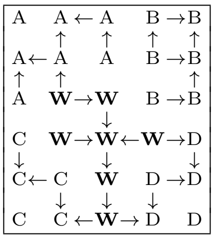
(f) LC-WT 2

A	A	A	B	B
A	A	C	B	B
C	C	C	B	B
C	C	C	C	D
C	C	C	C	D
C	C	C	D	D

(g) IFT-WT 1

A	A	B	B	B
A	A	D	B	B
D	D	D	B	B
D	D	D	D	D
C	D	D	D	D
C	C	D	D	D

(h) IFT-WT 2


(i) TD-WT and LCG

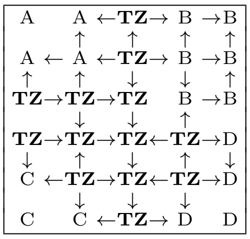

(j) TZ-IFT-WT and MOG

Figure 1. (a): Lower-complete input grayscale image with four minima. (b)–(j): Its WTs using 4-adjacency, according to definitions from literature. Label map is shown (**W** represents watershed line and **TZ** tie-zone) except for topological WT where watershed lines are valued. (c)–(h) show only two of the possible solutions. Watershed line in (b) and (i) is not separating. Arrows (pointing to predecessors) represent the lower complete graph (i) and multipredecessor optimal graph (j).

2.1 Watershed by image foresting transform (IFT-WT)

Under the IFT framework, an *image* is interpreted as a weighted graph $G = (V, A, w)$ consisting of a set V of *nodes* or *vertices* that represent image pixels, a set A of *arcs* weighted by w, a function from A to some nonnegative scalar domain. $N(v)$ denotes the *neighborhood* of node v, i.e., the set of nodes adjacent to it. Nodes u and v are adjacent when the arc $\langle u, v\rangle$ belongs to A. A graph (V', A') is *subgraph* of (V, A) if $V' \subseteq V$, $A' \subseteq A$ and $A' \subseteq V' \times V'$. A *forest* F of G is an acyclic subgraph F of G. *Trees* are connected components of the forest (any two nodes of a tree are connected by a path). A *path* $\pi(u, v)$ from node u to node v in graph (V, A, w) is a sequence $\langle u = v_1, v_2, \ldots, v_n = v\rangle$ of nodes of V such that $\forall i = 1 \ldots n-1$, $\langle v_i, v_{i+1}\rangle \in A$. A path is said *simple* if all its nodes are different from each other. A path with terminal node v is denoted by π_v. The path π_v is *trivial* when it consists of a single node $\langle v\rangle$. Otherwise, it can be defined by a path resulting from the concatenation $\pi_u \cdot \langle u, v\rangle$. A *path-cost function* f assigns to each path π a path cost $f(\pi)$, in some totally ordered set of cost values.

Let $S \subseteq V$ be a set of particular nodes s_i called *seeds*. For a given weighted graph (V, A, w) and a set S of seeds, the *image foresting transform* (IFT) returns a forest F of (V, A, w) such that (i) there exists for each node $v \in V$ a unique and simple path $\pi(s_i, v)$ in F from a seed node $s_i \in S$ to v and (ii) each such path is *optimum*, i.e., has a minimum cost for linking v to some seed of S, according to the specified path-cost function f. In other words, the IFT returns a shortest (cheapest in fact) path forest (SPF), also called *optimal forest* in this paper, where each tree is rooted to a seed. Although path costs are uniquely defined, the IFT may return many optimal forests because many paths of same minimum cost may exist for some nodes.

The *watershed transform by IFT* (IFT-WT) assumes that (i) the seeds correspond to regional minima of the image (or to imposed minima, i.e., markers [8]); (ii) the *max-arc* path-cost function $f_{\max}$ is used:

$$\begin{aligned} f_{\max}(\langle v\rangle) &= h(v), \\ f_{\max}(\pi_u \cdot \langle u, v\rangle) &= \max\{f_{\max}(\pi_u), w(u, v)\}, \end{aligned} \tag{1}$$

where $h(v)$ is a fixed but arbitrary handicap cost [16] for any paths starting at pixel v, and $w(u, v)$ is the weight of arc $\langle u, v\rangle \in A$, ideally higher on the object boundaries and lower inside the objects. Usual arc weight functions are: $w_1(u, v) = |I(u) - I(v)|$, $I(u)$ being the intensity of pixel u (cf. the so-called *watershed by dissimilarity* [15]); $w_2(u, v) = G(v)$, where $G(v)$ is the (morphological) gradient of image I at pixel v (cf. the *IFT-WT on gradient* [12, 15]). With this arc weight function, the max-arc path-cost function of Equation 1 can be simplified into: $f_{\max}(\langle v_1, v_2, \ldots, v_n\rangle) = \max\{G(v_1), G(v_2), \ldots, G(v_n)\}$. Note that the final cost map is unique and corresponds to the morphological superior reconstruction of the gradient image from the seeds using a flat structuring element. However, the forests and then the labelings may be multiple. Observe that a forest can be simply represented by a predecessor map P where $P(v)$ is the predecessor of node v in the minimum path. A label map L assigns to each node v the label $L(v)$ of the corresponding minimum-path root. The catchment basins correspond to the (labeled) trees: $CB_{IFT}(s_i) = \{v \in V, L(v) = L(s_i)\}$.

The so-called "plateau problem" is reported in WT literature for the internal non-minimum plateau pixels, i.e., non-minimum[1] pixels which have no lower neighbor. It can be solved by *lower completion* (cf. Definition 3.4 of [22]): a *lower complete image*[2] I_{LC} is computed from I by taking into account the geodesic distance of such internal pixels to the lower boundary of the plateau; then WT is applied on I_{LC}.

[1]Pixels which do not belong to regional minima.

[2]The improper term "image without plateau" is sometimes used instead.

In IFT-WT, $f_{lex} = (f_{\max}, f_d)$, a two-component *lexicographic cost* function, is used [15] to avoid a prior lower completion but has strictly the same role [4]. The first component, of highest priority, is the max-arc function representing the flooding process. The second one corresponds to the geodesic distance to the lower boundary of the plateau and makes different waters propagate on plateau at a same speed rate:

$$f_d(\langle v_1, \dots, v_n \rangle) = \max_{k \in [0,n-1]} \{k, f_{\max}(\langle v_1, \dots, v_n \rangle) = f_{\max}(\langle v_1, \dots, v_{n-k} \rangle)\}.$$

2.2 Tie zone

The choice of a single IFT-WT solution when many are possible is arbitrary and can be seen as a bias. Indeed, variations from one solution to another are sometimes significant and even unacceptable for some applications (e.g., reliable measures on segmented structures). In some images, an entire region is reached passing by a bottleneck pixel [2] and consequently included to the basin that first invades the bottleneck (like in Figure 1(g, h)). This problem is not related to the plateau problem and corresponds "to special pixel configurations which are not so rare in practice" as referred by [23].

It is why the *tie-zone concept* was proposed [4, 5] to unify the multiple solutions of a WT. Briefly speaking, considering all possible solutions derived from a specific WT definition, parts segmented in the same manner remain as catchment basins whereas differing parts are put in the tie zone (TZ). So, the TZ may be thick as well as empty.

In the case of IFT-WT, the *tie-zone watershed by IFT* (TZ-IFT-WS), returns a unique partition (cf. Figure 1(j)) of the image such that: A node is included in catchment basin $CB_{TZ-IFT}(s_i)$ when it is linked by a path to a same seed s_i in all the optimal forests (Φ denotes the set of the optimal forests F), otherwise it is included in the tie zone TZ:

$$CB_{TZ-IFT}(s_i) = \{v \in V, \quad \forall F \in \Phi, \quad \exists \pi(s_i, v) \text{ in } F\}, \qquad (2)$$
$$TZ_{IFT} = V \ \setminus \ \bigcup_i CB_{TZ-IFT}(s_i).$$

The area of the TZs, their distribution and number and distribution of their sources, the so-called bottlenecks, can be correlated with the robustness of a segmentation, i.e., with the degree of confidence a particular segmentation by WT has [2].

2.3 Multipredecessor optimal graph and lower complete graph

We introduce now a special graph, unique for each image, that will be used in Section 3. Roughly speaking, the *multipredecessor optimal graph* (MOG) of a weighted graph is the "union" of its optimal forests. More precisely, it is a directed acyclic subgraph of (V, A) such that its arc set A'' is the union of the (oriented) arcs of all the optimal forests $F \in \Phi$ (cf. Figure 1(j)):

$$MOG : (V, A'') = (V, \bigcup_{\forall F=(V,A')\in\Phi} A').$$

Once we have the lexicographical cost map of the image, i.e., a lower complete image, the following local property is valid: node p is predecessor of node v in the MOG if and only if p is neighbor of v with optimal lexicographic cost strictly lower than that of v (the superscript * denotes optimal paths).

$$\langle v, p\rangle \in A'' \quad \Leftrightarrow \quad p \in \mathbb{P}(v) \quad \Leftrightarrow \quad p \in N(v),\ f_{lex}(\pi_v^*) \succ f_{lex}(\pi_p^*), \tag{3}$$

where $\mathbb{P}(v)$ denotes the set of predecessors of node v, as the number of predecessors by node is no longer restricted to one as for the forests. Another property of the MOG is that if we independently choose one predecessor by non-minimum node, we obtain an optimal forest $(A' \subseteq A'' \subseteq A)$.

The *lower complete graph* (V, A''') (LCG, cf. Definition 3.5 of [22]) is analog to the MOG. Both are directed acyclic graphs built from the lower complete image. While all the *lower neighbors* in the lower complete image are predecessors of a node in the MOG, only the *steepest lower neighbors* are considered for a node in LCG (cf. Figure 1(i)).

$$\langle v, p\rangle \in A''' \Leftrightarrow p \in \mathbb{P}_{steepest}(v) \Leftrightarrow \begin{array}{l} p \in N(v),\ I_{LC}(v) > I_{LC}(p), \\ \frac{I_{LC}(v)-I_{LC}(p)}{d(v,p)} = \max_{q\in N(v)} \frac{I_{LC}(v)-I_{LC}(q)}{d(v,q)}, \end{array} \tag{4}$$

$d(p, q)$ being the distance between p and q. From Equation 3 and Equation 4, we deduce that $\mathbb{P}_{steepest}(v) \subseteq \mathbb{P}(v)$. Consequently, $A''' \subseteq A'' \subseteq A$ and the LCG (V, A''') of an image-graph (V, A) is a subgraph of its MOG (V, A'').

3. Watershed based on a local condition

As we said in Section 1, the *watershed transform based on a local condition* (LC-WT) is of "region" type because it has no watershed pixels [6, 9]. It may have multiple solutions (cf. Figure 1(e, f)). It assigns to each pixel the label of some minimum m_i, so as to form a partition of the image whose disjoint sets are the basins $CB_{LC}(m_i) = \{v \in V, L(v) = L(m_i)\}$.

As observed in refs. [6, 22], this WT definition is particularly well-suited for parallel implementations because it is based on a local condition. However, the overall WT computation is still a global operation. The meaning of locality in this definition is that one may subdivide an image in blocks, do a labeling of basins in each block independently, and make the results globally consistent in a final merging step.

Definition 1 (Watershed based on local condition)**.** For any lower complete image I_{LC}, a function L assigning a label to each pixel is called a watershed segmentation if:

1. $L(m_i) \neq L(m_j) \ \ \forall i \neq j$, with $\{m_k\}$ the set of minima of I_{LC};
2. for each pixel v with $\mathbb{P}_{steepest}(v) \neq \{\}, \exists p \in \mathbb{P}_{steepest}(v), L(v) = L(p)$.

□

The condition $\mathbb{P}_{steepest}(v) \neq \{\}$ means that v has at least one lower neighbor.

In other words, we can obtain a LC-WT by independently choosing one predecessor by non-minimum node in the precomputed LCG, and assigning a different basin label to each tree of the disjoint-set forest we obtained.

As the LCG (V, A''') generating such forests is a subgraph of the MOG (V, A'') generating any optimal forest, we conclude straightaway that these forests are optimal forests. Therefore: **any LC-WT is also an IFT-WT**.

4. Watershed based on topographic distance

We recall here the definition of *WT by topographic distance* (TD-WT) and some propositions from [18] for completeness.

Definition 2 (Watershed transform by topographic distance)**.** Let I be a gray-scale image, I_{LC} its lower completion, and $\{m_i\}$ the set of minima of I. Basin of I for minimum m_i and watershed are respectively:

$$\begin{aligned} CB_{TD}(m_i) &= \{v \in V, \ \ \forall j \neq i, \ I_{LC}(m_i) + T_{I_{LC}}(v, m_i) < I_{LC}(m_j) + T_{I_{LC}}(v, m_j)\} \\ W_{TD} &= V \ \setminus \ \bigcup_i CB_{TD}(m_i) \end{aligned} \tag{5}$$

□

$T_{I_{LC}}(p, q)$ being the *topographic distance* [18] between p and q:

$$T_{I_{LC}}(p,q) = \min_{\forall \pi(p,q)} T_{I_{LC}}^{\pi(p,q)}(p,q); \ T_{I_{LC}}^{\pi(p,q)}(p = p_1, q = p_n) = \sum_{i=1}^{n-1} cost(p_i, p_{i+1});$$

$$cost(p_i, p_{i+1}) = \begin{cases} LS(p_i)d(p_i, p_{i+1}), & \text{if } I_{LC}(p_i) > I_{LC}(p_{i+1}), \\ LS(p_{i+1})d(p_i, p_{i+1}), & \text{if } I_{LC}(p_i) < I_{LC}(p_{i+1}), \\ \frac{1}{2}\left[LS(p_i) + LS(p_{i+1})\right] d(p_i, p_{i+1}), & \text{if } I_{LC}(p_i) = I_{LC}(p_{i+1}). \end{cases}$$

The *lower slope* $LS(p)$ of I_{LC} at a pixel p is defined as the maximal slope linking p to any of its neighbors of lower altitude.

We call $(p_1, p_2, \ldots, p_n)$ a *path of steepest descent* from $p_1 = p$ to $p_n = q$ if $p_{i+1} \in \mathbb{P}_{steepest}(p_i)$ for $i = 1, \ldots, n-1$. A pixel p belongs to the *upstream* of q if there exists a path of steepest descent from p to q.

Proposition 1 (from [18]). *Let $I_{LC}(p) > I_{LC}(q)$. A path π from p to q is of steepest descent if and only if $T^{\pi}_{I_{LC}}(p,q) = I_{LC}(p) - I_{LC}(q)$. If a path π from p to q is not of steepest descent, $T^{\pi}_{I_{LC}}(p,q) > I_{LC}(p) - I_{LC}(q)$.*

This proposition implies that paths of steepest descent are the geodesics (shortest paths) of the topographical distance function. Consequently, from Definition 2 $CB_{TD}(m_i)$ is the set of points in the upstream of a single minimum m_i, i.e., there is (at least) one path of steepest descent to m_i and no path of steepest descent to any other minimum. The watershed consists of the points p which are in the upstream of at least two minima, i.e., there are at least two paths of steepest descent starting from p which lead to different minima.

4.1 Relationship with local-condition watershed

The forests representing the possible LC-WT generated from the LCG (Section 3) are made of paths of steepest descent. By strict analogy with Equation 2, we can conclude that: **TD-WT is the tie-zone transform of LC-WT**.

Proof. A node is included in catchment basin $CB_{TD}(m_i)$ when it is linked by a path to a same minimum m_i in all the forests made of steepest paths (the set of solutions for LC-WT, e.g., Figure 1(e, f)), otherwise it is included in the tie zone W_{TD} (cf. Figure 1(i)). As a consequence, we have also $CB_{TD}(m_i) \subseteq CB_{LC}(m_i)$ (cf. Figure 1(i)), as demonstrated in Theorem 2 of [6]. $\square$

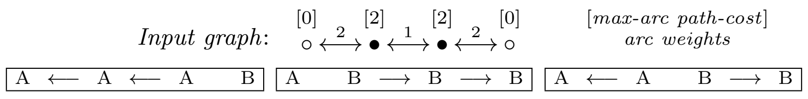
4.2 Relationship with tie-zone watershed by IFT

We saw in Section 3 that the set of LC-WT solutions is a subset of the set of IFT-WT solutions, so **the tie zone derived from the LC-WT solutions, i.e., W_{TD}, is a subset of TZ_{IFT}**: $W_{TD} \subseteq TZ_{IFT}$.

Proof. If pixel $p \in W_{TD}$, there are at least two paths of steepest descent from p to different minima. These paths belong to the LCG and to the MOG too (LCG is subgraph of MOG). So, there exist at least two optimal forests containing these paths leading to different minima. Consequently, $p \in TZ_{IFT}$. $\square$

Besides, **the catchment basins defined by TZ-IFT-WT are subsets of the corresponding basins defined by TD-WT**:
$\forall m_i, CB_{TZ-IFT}(m_i) \subseteq CB_{TD}(m_i)$.

Proof. If pixel $p \in CB_{TZ-IFT}(m_i)$, all the paths from p in the MOG lead to minimum m_i. So do the paths from p in the LCG (because LCG is subgraph of MOG, $\mathbb{P}_{steepest}(v) \subseteq \mathbb{P}(v), \forall v$). So, $p \in CB_{TD}(m_i)$. $\square$



(a) MSF 1 and SPF 1 **(b)** MSF 2 and SPF 2 **(c)** SPF 3 is not a MSF

Figure 2. A weighted graph with two markers (∘) and its 3 possible SPF-max and 2 MSF (total weight = 3). SPF 3 is not a MSF (total weight = 4).

5. Watershed based on a minimum spanning forest

The WT introduced in [17] is in fact a *WT from markers* (some significant minima are selected to avoid oversegmentation). It uses a weighted *neighborhood graph* whose nodes are the primitive catchment basins corresponding to regional minima of the image. Arcs are placed between neighbor catchment basins and weighted by the altitude of the *pass* between them. A *watershed based on minimum spanning forest* (MSF-WT) is defined on this weighted graph: the many possible MSFs on the graph define partitions that are considered solutions of this WT. Each tree of the MSF is a catchment basin of the MSF-WT.

A tree (V, T) is a *minimum spanning tree* (MST) of graph (V, A, w) if its total weight $\sum_{t \in T} w(t)$ (sum of the weight of its arcs) is minimum. It is unique when all the arc weights of the graph are different. A *minimum spanning forest* (MSF) is a forest whose total weight (sum of the weight of its arcs) is minimum and where each node is linked to a seed $s_i \in S$ by a unique simple path. The MSF problem for weighted graph (V, A, w) can be solved by constructing the MST of (V^*, A^*, w^*) where a ficticious root node z and arcs of weight -1 linking z to each seed were added. In a final step, these negative arcs will be removed to obtain a MSF.

Theorem 1 (Minimum spanning tree [13])**.** *(V, T) is a tree of minimum weight for graph (V, A, w) if and only if for every arc $u \in A - T$ the cycle μ^u (such that $\mu^u \subset T + \{u\}$) satisfies: $w(u) \geq w(v)$, $\forall v \in \mu^u$ $(v \neq u)$.*

Now, we demonstrate[3] that the set of MSF solutions is a subset of the set of IFT-WT solutions defined by the same weighted graph using the same seed set with seed handicaps $h(s_i) = 0$ and the max-arc path cost[4].

Theorem 2 (Shortest-path forest and minimum spanning forest)**.**
Given a weighted graph and a seed set, any minimum spanning forest (MSF) is also a shortest-path forest (SPF-max) using max-arc path cost $f_{\max}$.

F is a MSF $\Rightarrow$ F is a SPF-max (or IFT-WT).

[3]This result was obtained independently in [1].
[4]Until now, lexicographic path-cost $f_{lex} = (f_{\max}, f_d)$ was used for IFT-WT.

Reciprocal is false (cf. examples and counter-example in Figure 2).

Proof. Suppose that F is a MSF and T the corresponding MST using a ficticious root z. Suppose that there exists a path π from p to z, π belongs to T and π is non optimal in the SPF-max sense (i.e., using $f_{\max}$). Suppose that there exists another path π' from p to z such that $f_{\max}(\pi') < f_{\max}(\pi)$. Then for every arc v in π', its weight $w(v) \leq f_{\max}(\pi') < f_{\max}(\pi)$. Now, there exists an arc u in π', u not in T (because T has no cycle: p and z are linked by only one simple path). Therefore, $w(u) < f_{\max}(\pi)$. Now, T is a MST. Therefore, from Theorem 1, $w(u) \geq f_{\max}(\pi \cdot \pi') \geq f_{\max}(\pi)$, $\pi \cdot \pi'$ being the cycle μ^u formed by concatenation of the two paths. That is a contradiction with the previous conclusion. So, any MSF is necessarily SPF-max. □

6. Conclusion and future works

In this paper, we used the IFT-WT and the TZ concept (that unifies the set of multiple solutions of a given WT) to relate some discrete WT definitions and, thereby, better understand the differences between the multiple solutions given by such definitions. We demonstrate that (i) the TD-WT corresponds to the tie-zone transform of the LC-WT; (ii) the possibly thick and not separating watershed line of TD-WT is contained in the TZ of the TZ-IFT-WT (with lexicographic cost function), while (iii) the catchment basins of the former contain the basins of the latter; (iv) any solution of LC-WT is also solution of the IFT-WT; (v) any solution of MSF-WT is also solution of the IFT-WT (with max-arc path-cost function).

We are preparing an extended version of this paper which will also include the comparative analysis of flooding-WT definition with IFT-WT and TZ, as well as some issues on related algorithms.

Acknowledgments

This work is supported by CAPES.

References

[1] C. Allène, J.-Y. Audibert, M. Couprie, J. Cousty, and R. Keriven, *Some links between min cuts, optimal spanning forests and watersheds*, Procs. 8th International Symposium on Mathematical Morphology, October 2007.

[2] R. Audigier and R. A. Lotufo, *Tie-Zone Watershed, Bottlenecks and Segmentation Robustness Analysis*, XVIII Brazilian Symp. on Computer Graph. and Image Proc. (SIBGRAPI'05), October 9–12, 2005, pp. 55–62.

[3] ———, *Duality between the Watershed by Image Foresting Transform and the Fuzzy Connectedness Segmentation Approaches*, XIX Brazilian Symp. on Computer Graph. and Image Proc. (SIBGRAPI'06), October 2006, pp. 53–60.

[4] ______, *Uniquely-Determined Thinning of Tie-Zone Watershed Based on Label Frequency*, Journal of Mathematical Imaging and Vision **27** (2007), no. 2, 157–173.

[5] R. Audigier, R. A. Lotufo, and M. Couprie, *The Tie-Zone Watershed: Definition, Algorithm and Applications.*, Proceedings of IEEE International Conference on Image Processing (ICIP'05), September 11–14, 2005, pp. 654–657.

[6] A. Bieniek, H. Burkhardt, H. Marschner, M. Nölle, and G. Schreiber, *A Parallel Watershed Algorithm*, Procs. 10th Scandinavian Conference on Image Analysis (SCIA'97), June 1997, pp. 237–244.

[7] S. Beucher and C. Lantuéjoul, *Use of Watersheds in Contour Detection*, International Workshop on Image Processing, Real-Time Edge and Motion Detection/Estimation, 1979.

[8] S. Beucher and F. Meyer, *The Morphological Approach to Segmentation: The Watershed Transform* (E. R. Dougherty, ed.), Marcel Dekker, Inc., New York (NY), USA, 1993, Mathematical Morphology in Image Processing, pp. 433–481.

[9] A. Bieniek and A. Moga, *A connected component approach to the watershed segmentation*, Mathematical Morphology and its Applications to Image and Signal Processing, 1998, pp. 215–222.

[10] M. Couprie and G. Bertrand, *Topological Grayscale Watershed Transformation*, SPIE Vision Geometry V Proceedings, 1997, pp. 136–146.

[11] A. X. Falcão, B. S. Cunha, and R. A. Lotufo, *Design of Connected Operators using the Image Foresting Transform*, SPIE on Medical Imaging **4322** (2001), 468–479.

[12] A. X. Falcão, J. Stolfi, and R. A. Lotufo, *The Image Foresting Transform: Theory, Algorithms, and Applications*, IEEE Transactions on Pattern Analysis and Machine Intelligence **26** (2004), no. 1, 19–29.

[13] M. Gondran and M. Minoux, *Graphs and algorithms*, John Wiley & Sons Ltd., 1984.

[14] Y. C. Lin, Y. P. Tsai, Y. P. Hung, and Z. C. Shih, *Comparison Between Immersion-Based and Toboggan-Based Watershed Image Segmentation*, IEEE Transactions on Image Processing **15** (2006), no. 3, 632–640.

[15] R. A. Lotufo and A. X. Falcão, *The Ordered Queue and the Optimality of the Watershed Approaches*, Procs. 5th International Symposium on Mathematical Morphology, June 26–28, 2000, pp. 341–350.

[16] R. A. Lotufo, A. X. Falcão, and F. A. Zampirolli, *IFT-Watershed from Gray-Scale Marker*, XV Brazilian Symp. on Computer Graph. and Image Proc. (SIBGRAPI'02), October 2002, pp. 146–152.

[17] F. Meyer, *Minimum spanning forests for morphological segmentation*, Procs. 2nd International Symposium on Mathematical Morphology, 1994, pp. 77–84.

[18] ______, *Topographic distance and watershed lines*, Signal Processing **38** (1994), 113–125.

[19] L. Najman and M. Schmitt, *Watershed of a continuous function*, Signal Processing **38** (1994), no. 1, 99–112.

[20] V. O. Ruiz, J. I. G. Llorente, N. S. Lechon, and P. G. Vilda, *An improved watershed algorithm based on efficient computation of shortest paths*, Pattern Recognition **40** (2007), no. 3, 1078–1090.

[21] F. Prêteux, *On a distance function approach for gray-level mathematical morphology* (E. R. Dougherty, ed.), Marcel Dekker, Inc., New York (NY), USA, 1993, Mathematical Morphology in Image Processing, pp. 323–349.

[22] J. B. T. M. Roerdink and A. Meijster, *The Watershed Transform: Definitions, Algorithms and Parallelization Strategies*, Fundamenta Informaticae **41** (2000), 187–228.

[23] L. Vincent and P. Soille, *Watersheds in digital spaces: An efficient algorithm based on immersion simulations*, IEEE Transactions on Pattern Analysis and Machine Intelligence **13** (1991), no. 6, 583–598.

Watershed-based visualization of high-density EEG coherence

Michael ten Caat[1, 2], Natasha M. Maurits[2, 3] and Jos B. T. M. Roerdink[1, 2]

[1] *Institute for Mathematics and Computing Science, University of Groningen, The Netherlands*
{mtc@cs.rug.nl,j.b.t.m.roerdink}@rug.nl

[2] *BCN Neuroimaging Center, University of Groningen, The Netherlands*

[3] *Department of Neurology, University Medical Center Groningen, University of Groningen, The Netherlands*
n.m.maurits@neuro.umcg.nl

Abstract Synchronous electrical activity in different brain regions is generally assumed to imply functional relationships between these regions. A measure for this synchrony is electroencephalography (EEG) coherence. Recently, we developed a new method for data-driven visualization of high-density EEG coherence, avoiding the visual clutter of conventional data-driven methods. It uses the concept of maximal clique detection, having time complexity $O(3^{n/3})$ with n the number of vertices. Here, a more efficient clustering method is used with time complexity $O(n^2 \log n)$, based on a watershed algorithm which is modified to detect cliques in a greedy way. Here, it obtains a speedup of factor 400 while results are similar, making interactive visualization of high-density EEG coherence feasible.

Keywords: EEG, coherence, graph, watershed transform, medical data.

1. Introduction

Electroencephalography (EEG) measures the electrical activity of the brain using electrodes attached to the scalp at multiple locations. Synchronous electrical activity in different brain regions is generally assumed to imply functional relationships between these regions. A measure for this synchrony is EEG coherence [6,9]. Visualization of high-density EEG (at least 64 electrodes) is not always managed well [12–14]. For the analysis of high-density EEG coherence, EEG researchers often employ a hypothesis-driven definition of certain regions of interest (ROIs). In these ROIs, all electrodes are assumed to record similar signals, because of volume conduction effects [8]. As an alternative to the hypothesis-driven approach, we previously introduced a data-driven visualization of ROIs which shows less clutter than conventional data-driven methods [14]. It is based on a graph with vertices representing electrodes and edges representing significant coherences between electrode signals. The data-driven version of a ROI is called a functional unit (FU) and is represented in the graph by a clique consisting of spatially connected vertices [14].

Our existing maximal clique based (MCB) method clusters vertices into FUs using the concept of maximal clique detection [2], having time complexity $O(3^{n/3})$ with n the number of vertices [15], which we extended to find sets of spatially connected vertices [14]. With an interactive visualization of EEG coherence in mind, we here present a new method with time complexity $O(n^2 \log n)$, based on the concept of the watershed transform [11] which is adapted to detect cliques in a greedy way. We refer here to the more efficient new method as watershed based (WB) method.

2. Preliminaries

2.1 EEG data

EEG can be recorded using up to 512 electrodes attached to the scalp. A conductive gel is applied between skin and electrodes to reduce impedance. The electrical potential is measured at all electrodes simultaneously. The measured signals are amplified, resulting in one recording channel for every electrode. If there are many electrodes, the term 'multichannel' or 'high-density' EEG is used. As a result of volume conduction [8], multiple electrodes can record a signal from a single source in the brain. Therefore, nearby electrodes usually record similar signals. Because sources of activity at different locations may be synchronous, electrodes far apart can also record similar signals. A measure for this synchrony is coherence, calculated between pairs of signals as a function of frequency. The coherence c_λ as a function of frequency λ for two continuous time signals x and y is defined as the absolute square of the cross-spectrum f_{xy} normalized by the autospectra f_{xx} and f_{yy} [6], having values in the interval $[0,1]$: $c_\lambda(x,y) = \frac{|f_{xy}(\lambda)|^2}{f_{xx}(\lambda) f_{yy}(\lambda)}$. An event-related potential (ERP) is an EEG recording of the brain response to a sensory stimulus. For L repetitive stimuli, the EEG data can be separated into L segments, each containing one brain response. A *significance threshold* for the estimated coherence is then given by [6]

$$\varphi = 1 - p^{1/(L-1)}, \tag{1}$$

where p is a probability value associated with a confidence level α, such that $p = 1 - \alpha$. Throughout this paper, we use $p = 0.05$, unless stated otherwise.

2.2 Graph theory

A graph $G = (V, E)$ consists of a set of vertices V and a set of edges $E \subseteq V \times V$. The vertices u and v are called neighbors or adjacent if there is one edge between them. The neighborhood of vertex v is the collection of all neighbors of v. In a directed graph, the set E consists of ordered pairs of vertices from V. In an undirected graph, the pairs are not ordered. A directed edge is denoted as $e = (u, v)$, an undirected edge as $e = \{u, v\}$; u and v are called incident with e, and e is said to be incident with u and v. A walk between two vertices is a sequence of edges $(e_1, ..., e_n)$, with vertices $v_0, ..., v_n$ such that $e_i = \{v_{i-1}, v_i\}$. If a walk exists between two vertices, they are called connected. For a graph $G = (V, E)$ and $V' \subseteq V$, the set of all edges with both vertices in V' is denoted as $E|V'$. The graph $G' = (V', E|V')$ is called the (vertex-) induced subgraph on V'. If $V' \subset V$ and $E' \subset E|V'$, then $G' = (V', E')$ is called a subgraph. If any two vertices in $G = (V, E)$ are connected, G is called a connected graph. A maximal connected subgraph of G is a connected component. If all two-element subsets of V are edges, then $G = (V, E)$ is a complete graph. A clique is a set $V' \subseteq V$ such that the induced subgraph on V' is a complete graph. A maximal clique is a clique which is not a subgraph of a larger clique. For more background information on graphs, see, e.g., [7].

3. Data representation

3.1 Experimental setup

Here, brain responses from three young adults are studied, recorded using an EEG cap with 119 scalp electrodes. During a so-called P300 experiment, each participant was instructed to count target tones of 2000 Hz (probability 0.15), alternated with standard tones of 1000 Hz (probability 0.85) which were to be ignored. After the experiment, the participant had to report the number of perceived target tones. For each dataset, brain reactions to 20 target tones were recorded in $L = 20$ segments.

A procedure from Neurospec (`www.neurospec.org`) was adopted to compute the coherence. Frequencies between 1 and 30 Hz are typically studied clinically. We calculated the coherence within a lower (1-3 Hz) and a higher (13-20 Hz) EEG frequency band, because EEG synchrony behaves differently for lower and higher frequencies [9, 10]. For 119 electrodes, in total 7021 coherence values were computed per frequency band. If the conductive gel accidentally connected two adjacent electrodes, very high coherences were measured. Coherences higher than 0.99 were therefore ignored.

3.2 EEG coherence graph

The data are represented by a *coherence graph* with vertices representing electrodes. Coherences above the significance threshold (Equation 1) are represented by edges, coherences below the threshold are ignored.

To determine spatial relationships between electrodes, a Voronoi diagram is employed which partitions the plane into regions with the same nearest vertex. For EEG data, the vertex set equals the set of electrode positions (Figure 1). The vertices are referred to as (Voronoi) centers, the boundaries as (Voronoi) polygons. The area enclosed by a polygon is called a (Voronoi) cell. We call two cells *Voronoi neighbors* if they have a boundary in common. A collection of cells C is called *Voronoi-connected* if for a pair $\varphi_0, \varphi_n \in C$ there is a sequence $\varphi_0, \varphi_1, ..., \varphi_n$ of cells in C with each pair φ_{i-1}, φ_i consisting of Voronoi neighbors. We use the terms "Voronoi neighbor" and "Voronoi-connected" interchangeably for cells, vertices, and electrodes.

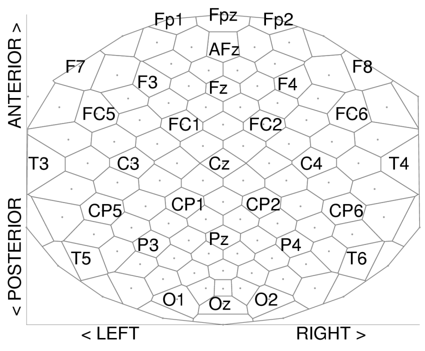


Figure 1. Voronoi diagram, with (a subselection of) all electrode labels in the corresponding cells (top view of the head, nose at the top). To improve the readability, the Voronoi diagram is stretched horizontally. The boundary is the convex hull of all electrodes. Because the coherence computation is independent of distance, distances between electrodes do not need to be preserved. However, spatial relationships between electrodes are maintained.

4. Vertex clustering

4.1 Existing maximal clique based (MCB) method

A *functional unit* (FU) is a clique consisting of Voronoi-connected vertices [14]. Consequently, an FU corresponds to a Voronoi-connected set of electrodes in which the electrodes record pairwise significantly coherent signals. Our existing FU clustering method for high-density EEG coherence [14] uses maximal clique detection [2], which we extended to find sets of Voronoi-connected vertices. Every vertex can be part of multiple (Voronoi-connected) maximal cliques. To assign a unique label to every vertex, a quantity *total strength* is defined for a (sub)graph $G = (V, E)$ as the sum of all edge values. This value is not normalized for the size of E. Consequently, if two graphs have an equal average coherence, the graph with more vertices has a higher total strength. Next, all cliques are queued in decreasing order by their total strength. Then the following labeling procedure is repeated, until there are no more cliques or until all vertices are labeled. The first clique is removed from the queue, and all its vertices are assigned a unique label and are removed from the other cliques. If necessary, the changed cliques are separated into Voronoi-connected components. For all changed cliques, the total strength is recomputed before they are put in the appropriate position in the sorted queue. After completion of the labeling procedure, every set of identically labeled vertices is an FU.

The worst-case time complexity of maximal clique detections is $O(3^{n/3})$, with n the number of vertices [15]. In practice, performance of maximal clique detection strongly depends on graph structure [16].

4.2 New watershed based (WB) method

As an alternative to the MCB method, we present here a greedy method approximating maximal cliques on the basis of the watershed transform [11]. In the usual watershed algorithm, a subset of all local minima is selected as markers. Markers are labeled and are associated with basins. Basins contain vertices with the same label as the corresponding marker and are extended as follows, using the watershed implementation based on ordered queues [1]. The first vertex v is removed from a queue of vertices sorted in decreasing order of priority. Every unlabeled neighbor v' of v receives the same label as v and is put into the queue with a priority depending on the value of v', but not higher than the priority of v. This continues until the queue is empty.

Now we modify the usual watershed transform in order to obtain spatially connected sets of electrodes, where all electrodes in a given set have recorded mutually significantly coherent signals. This modification concerns two steps in the watershed transform: (i) choice of markers; (ii) use of an edge queue instead of a vertex queue. We explain these two points in more detail.

First, we define a marker as an electrode recording a signal that is locally maximally coherent with signals of its spatially neighboring electrodes. Because the EEG coherence graph has edge values instead of vertex values, we first assign a coherence value to each vertex by computing the average of the edge values between this vertex and all its Voronoi neighbors. Then, we select all vertices which are local maxima as markers to be associated with basins, because those vertices are locally maximally similar to their spatially neighboring vertices. Note that we choose *all* local maxima as markers, instead of a small subset as is usually done when the watershed algorithm is applied to digital images. In our case the over-segmentation problem is less severe, because the number of electrodes is an order of magnitude smaller than the number of pixels in an image. If the number of basins (i.e., clusters) found is still too large, we can suppress basins below a certain size in a post-processing step (see Section 6).

The second point concerns the type of queue we use. Whereas the usual queue-based implementation of the watershed transform uses a vertex queue sorted in increasing order of gray value, we use an edge queue sorted in decreasing order of coherence value. (The vertex values are only used for defining the markers, as indicated above.) In case the coherence graph has multiple identical edge values (which did not occur for our datasets), an ordered queue consisting of queues with identically valued elements can be used, as for digital images which usually contain multiple identically valued vertices [1].

This queue is initialized with edges (corresponding with a significant coherence) between markers and their Voronoi neighbors. The first edge (v, v') in the queue corresponds to the highest similarity (coherence) between any vertex v' outside and a Voronoi neighboring vertex v inside a basin. Therefore, vertex v' is the first candidate to be added to a basin.

The greedy WB method maintains the following dynamic vertex sets for the detection of Voronoi-connected cliques.

- bsn_i contains a sorted list of the vertices in basin i.
- $L(v)$ contains the basin label of vertex v.
- *adjCohBsn*$_i$ contains a list of vertices (sorted by vertex number) which are adjacent to *each* of the vertices in bsn_i in the coherence graph.
- *queue* contains edges in decreasing order. When vertex v receives a label, an edge $e = (v, v')$ is added to *queue* for each unlabeled Voronoi neighbor v' of v, provided that the corresponding edge value exceeds the significance threshold (Equation 1).

The main procedure consists of the following steps. Remove the first edge, say $e = (v, v')$ from *queue*. In case vertex v' was also labeled between the insertion and removal of $e = (v, v')$, nothing is done and the procedure continues with a new edge. Otherwise (v' is unlabeled), there are two cases. In case $v' \in adjCohBsn_{L(v)}$, v' receives label $L(v)$ and v' is added to $bsn_{L(v)}$; $adjCohBsn_{L(v)}$ is replaced by its intersection with the neighborhood of v' in the coherence graph; *queue* is extended with the edges between v' and its Voronoi-neighbors, provided that corresponding edge values exceed the significance threshold. In the other case, if $v' \notin adjCohBsn_{L(v)}$, v' is not labeled (yet). This procedure is repeated until *queue* is empty. Each basin then corresponds to an FU.

The WB vertex clustering procedure is described below in pseudocode. The operation *insertEdgeSort(e(v, v'),queue)* inserts edge $e(v, v')$ into the appropriate position in a decreasingly sorted edge queue *queue*; similarly, *insertVSort(v,vqueue)* inserts vertex v into the appropriate position in a decreasingly sorted vertex queue *vqueue*; *dequeue(queue)* returns and removes the first edge from an edge queue *queue*; *intersect(.,.)* gives the intersection of two sorted vertex sets. The size of a vertex set is denote by $| \,.\, |$.

The creation of the sorted edge queue (step 1) has time complexity $O(m \log m) = O(n^2 \log n)$, where n denotes the number of vertices and $m = \frac{n(n-1)}{2}$ the maximal number of edges. Then (step 2), for every edge $e = (v, v')$ in the queue with unlabeled v' (queue length $O(m) = O(n^2)$), the following steps are executed consecutively: (a) binary search ($O(\log n)$) is used to verify the presence of v' in the set $adjCohBsn_{L(v)}$, a sorted set of at most n vertices; (b) two sorted vertex sets (with maximum size n)

Algorithm 1 WB pseudocode.

```
INPUT: $V$ is the vertex set; $marker(i)$ = marker $i$;
    $c(v, v') = \text{coherence}(v, v') = c(v', v)$; $\varphi$ = sign. threshold;
    $adjCoh_v = \{v' \in V \mid c(v,v') \geq \varphi\}$;
    $adjVor_v = \{v' \in V \mid v' \in \text{Vor.-neighbors}_v \ \& \ v' \in adjCoh_v\}$;
    $\{adjCoh_v,\ adjVor_v$ are both sorted by vertex number$\}$
OUTPUT: $bsn_i$ is basin $i$ (i.e., an FU) sorted by vertex number
INITIALIZATION:
 1: $queue \leftarrow \emptyset$ {queue of edges}
 2: for all $v \in V$ do
 3:   $L(v) \leftarrow 0$ {$L(v)$ = label of vertex $v$}
 4: end for
 5: for $i = 1$ to $|marker|$ do
 6:   $bsn_i \leftarrow marker(i)$; $v \leftarrow marker(i)$; $L(v) \leftarrow i$
 7:   $adjCohBsn_{L(v)} \leftarrow adjCoh_v$
 8:   for all $v' \in adjVor_v$ do
 9:     $insertEdgeSort(e(v,v'),queue)$
10:   end for
11: end for
MAIN:
12: while $queue \neq \emptyset$ do
13:   $e(v, v') \leftarrow dequeue(queue)$
14:   if $L(v') = 0$ then
15:     if $v' \in adjCohBsn_{L(v)}$ then
16:       $adjCohBsn_{L(v)} \leftarrow intersect(adjCohBsn_{L(v)}, adjCoh_{v'})$
17:       $L(v') \leftarrow L(v)$; $bsn_{L(v)} \leftarrow insertVSort(v', bsn_{L(v)})$
18:       for all $v^* \in adjVor_{v'}$ do
19:         if $L(v^*) \neq 0$ then
20:           $insertEdgeSort(e(v', v^*), queue)$
21:         end if
22:       end for
23:     end if
24:   end if
25: end while
```

are intersected ($O(n)$); (c) Voronoi-neighbors of v (at most n) are inserted into the edge queue ($O(n \log m) = O(n \log n)$). Step c has a higher time complexity than a and b. However, step b and c are only executed $O(n)$ times, because at most n vertices can be added to a basin. Thus, the time complexity for step 2 is $O(n^2 \log n)$, as for step 1, which is therefore the total time complexity.

5. FU map

In a so-called *FU map*, each FU is visualized as a set of identically colored Voronoi cells, with different colors for adjacent FUs [14]. Given the FUs, the *inter-FU coherence* c' at frequency λ between two functional units W_1 and W_2 is defined as the sum of the coherence values between one vertex in W_1 and the other vertex in W_2, scaled by the total number of edges between W_1 and W_2 [14]:

$$c'_\lambda(W_1, W_2) = \frac{\sum_{i,j}\{c_\lambda(v_i, v_j) \mid v_i \in W_1, v_j \in W_2\}}{|W_1| \cdot |W_2|}, \tag{2}$$

where, $|W_i|$ indicates the number of vertices in W_i. Note that coherences between *any* pair of vertices are taken into account, to normalize for the size of the FUs. A line is drawn between FU centers if the corresponding inter-FU coherence exceeds a threshold. We consistently choose this threshold to be equal to the significance threshold (Equation 1), because we already used this threshold to determine the coherence graph.

6. Results

We show FU maps for the MCB (Figure 2) and the WB (Figure 3) method, for three datasets and two frequency bands. Because FU maps including small FUs fail to give a good overview [14], only FUs larger than 5 cells are considered.

MCB FU maps (Figure 2) were previously shown to agree with earlier findings in the literature [14]. The number of connecting lines between FUs was lower for a higher EEG frequency, in accordance with [9, 10]. Furthermore, connections between anterior and posterior FUs were probably associated with the two most important sources of brain activity for this data type [3, 4]. Here, the WB FU maps (Figure 3) confirmed these findings.

Each WB FU map was compared to the corresponding MCB FU map (compare Figure 3 to Figure 2). From a visual inspection, the biggest dissimilarity appeared between the FU maps for *dataset 2* and frequency band *1-3 Hz* (Figure 2 and Figure 3: top row, middle). For this case, the WB method showed more FUs (at positions where the MCB method has no sufficiently large FUs) and more inter-FU connections than the MCB method. However, the six inter-FU connections with the highest value in the WB

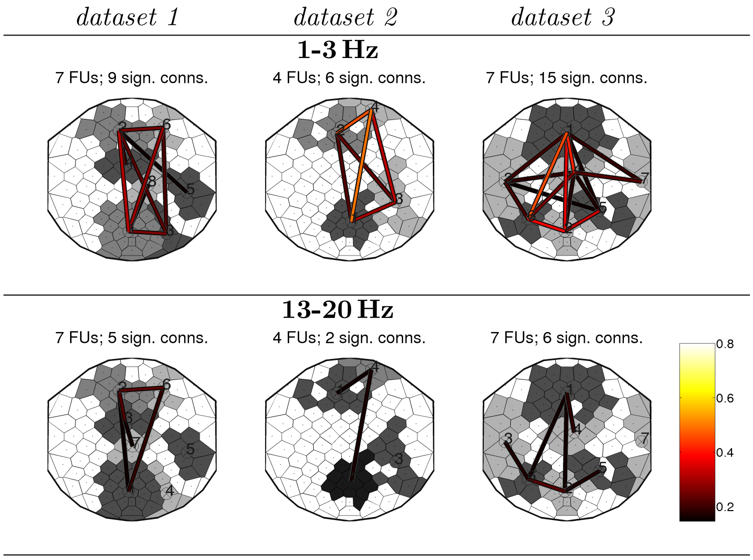


Figure 2. **MCB FU maps** (top view of the head, nose at the top) with FUs larger than 5 cells, for EEG frequency 1-3 Hz (*top row*) and 13-20 Hz (*bottom row*), for three datasets and $p = .05$. Each FU is visualized as a set of identically colored Voronoi cells, with different gray values for adjacent FUs. White Voronoi cells are part of FUs with $|\text{FU}| \leq 5$. Geographic centers of FUs are visualized as a circle with a cross inside, having a color corresponding to the FU. A line connects FUs if the inter-FU coherence exceeds the significance threshold (Equation 1), with its color depending on the value (see color bar, with minimum corresponding to the coherence threshold ≈ 0.15). Lines are drawn in the order from low to high inter-FU coherence values. Above each FU map the number of FUs and the number of connecting lines between FUs are displayed.

FU map corresponded to the six inter-FU connections in the MCB FU map. This became more clear by simultaneously increasing the significance threshold for both the coherence graph and the inter-FU connections in the WB FU map (i.e., decreasing the p-value). The result showed only the six highest inter-FU connections for $p = .001$ (Figure 4, right). Those six connecting lines completely connected two anterior and two posterior FUs for the WB method (Figure 4, right), which were in orientation very similar to the six connecting lines for the MCB method (Figure 4, left). The two anterior FUs in the MCB FU map did not correspond exactly with the two anterior FUs obtained with the WB method, because the latter is a greedy FU detection method. The same holds for the posterior FUs.

For the other cases, inter-FU connections between FUs which were no spatial neighbors were usually similar for the MCB and WB method. Further, the locations of the FUs were usually similar for the MCB and WB method, especially for FUs connected with another FU which was not a

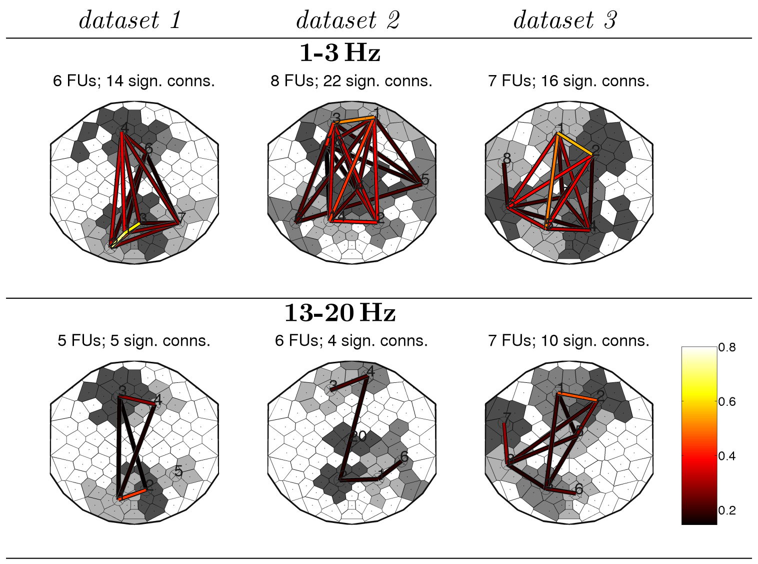


Figure 3. **WB FU maps**, with FUs larger than 5 cells, for the 1-3 Hz EEG frequency band (*top row*) and for 13-20 Hz (*bottom row*), for three datasets and $p = .05$.

spatial neighbor. For example, anterior and posterior FUs connected by a line were similarly located for all cases.

Additionally, we compared the numbers of FUs obtained with both methods. In the range up to an FU size of 5 cells (not illustrated), the MCB method had on average 14 FUs and the WB method 6 FUs. In the range

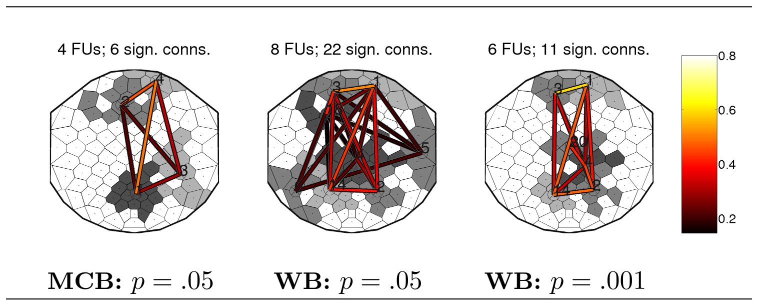


Figure 4. FU maps for *dataset 2* and frequency band *1-3 Hz*, $|FU| > 5$. **Left:** MCB, $p = .05$. **Middle:** WB, $p = .05$. **Right:** WB, $p = .001$. The WB FU map is more similar to the MCB FU map (left) when the the significance threshold (Equation 1) is increased (corresponding to decreasing p) for the WB method (right).

above an FU size of 5 cells, both methods had on average 6 FUs (Figure 2 and Figure 3). Thus, the maximal clique method detected relatively more small-size FUs.

For the FU maps created with both methods in Figure 2 and Figure 3, we determined the CPU time consumed by FU detection. On average, the (non-optimized) MCB and WB methods took 24 s and 0.06 s, respectively, meaning that a speedup of factor 400 was obtained by the new WB method.

7. Discussion and conclusions

EEG coherence analysis is defined as the study of coherence between functional units (FUs). We previously introduced the maximal clique based (MCB) method for data-driven visualization of high-density EEG coherence [14], avoiding the visual clutter of conventional data-driven visualizations. This MCB method makes use of the concept of maximal clique detection with time complexity $O(3^{n/3})$ for n electrodes. In practice, performance of maximal clique detection strongly depends on graph structure [16]. With an interactive visualization in mind, we have designed here a new watershed based (WB) method having time complexity $O(n^2 \log n)$, based on the watershed transform which is modified to detect cliques in a greedy way. The existing MCB and the new WB method both detect data-driven ROIs represented by cliques consisting of spatially connected vertices, referred to as FUs.

Comparing the MCB and the WB method, the greedy WB method directly results in uniquely labeled electrodes, contrary to the MCB method. The existing MCB method shows a relatively larger number of smaller FUs than the new WB method. The MCB and the WB method both depend on three thresholds. The first two thresholds concern the initial coherence graph and the inter-FU coherence. Both were chosen to be equal to the significance threshold. The third threshold concerns the minimal FU size. FUs and inter-FU connections were usually similar for the MCB and the greedy WB method. If not, systematically adapting the significance threshold revealed a strong similarity between FU maps obtained with both FU detection methods. Thus, both methods visualize similar information. Further, this information was found to agree with conventional findings [3, 4, 9, 10].

The watershed transform is generally known to suffer from over-segmentation, for which a solution may be based on the concept of dynamics [5]. However, dynamics are defined for vertex values, whereas the EEG coherence graph has edge values. For EEG coherence, there are two straightforward solutions for potential over-segmentation. First, two spatially neighboring FUs may be merged if the union of the two corresponding vertex sets is a clique. Second, determination of the markers can be based not only on (first degree) Voronoi neighbors, but also on (second degree) Voronoi neighbors of Voronoi neighbors.

Here, the new WB method is up to a factor 400 faster than the existing MCB method. The new method makes interactive visualization of high-density EEG coherence feasible for the intended users, including EEG researchers and clinical experts.

References

[1] S. Beucher and F. Meyer, *The morphological approach to segmentation: the watershed transformation*, Mathematical Morphology in Image Processing, 1993, pp. 433–481.

[2] C. Bron and J. Kerbosch, *Algorithm 457: finding all cliques of an undirected graph*, Commun. ACM **16** (1973), no. 9, 575–577.

[3] M. D. Comerchero and J. Polich, *P3a and P3b from typical auditory and visual stimuli*, Clin. Neurophysiol. **110** (1999), no. 1, 24-30.

[4] J. W. Elting, T. W. van Weerden, J. van der Naalt, J. H. A. De Keyser, and N. M. Maurits, *P300 Component Identification Using Source Analysis Techniques: Reduced Latency Variability*, Journal of Clinical Neurophysiology **20** (2003), no. 1, 26–34.

[5] M. Grimaud, *A new measure of contrast: dynamics*, SPIE: Image Algebra and Morphological Processing III, 1992, pp. 292–305.

[6] D. M. Halliday, J. R. Rosenberg, A. M. Amjad, P. Breeze, B. A. Conway, and S. F. Farmer, *A framework for the analysis of mixed time series/point process data - Theory and application to the study of physiological tremor, single motor unit discharges and electromyograms*, Progr. Biophys. Mol. Biol. **64** (1995), no. 2/3, 237–278.

[7] D. Jungnickel, *Graphs, Networks and Algorithms*, Springer, 1999.

[8] J. P. Lachaux, E. Rodriguez, J. Martinerie, and F. J. Varela, *Measuring phase synchrony in brain signals.*, Hum Brain Mapp **8** (1999), no. 4, 194-208.

[9] N. M. Maurits, R. Scheeringa, J. H. van der Hoeven, and R. de Jong, *EEG coherence obtained from an auditory oddball task increases with age.*, J. Clin. Neurophysiol. **23** (2006), no. 5, 395-403.

[10] P. L. Nunez, R. Srinivasan, A. F. Westdorp, R. S. Wijesinghe, D. M. Tucker, R. B. Silberstein, and P. J. Cadusch, *EEG coherency. I: Statistics, reference electrode, volume conduction, Laplacians, cortical imaging, and interpretation at multiple scales.*, Electroencephalogr. Clin. Neurophysiol. **103** (1997), no. 5, 499-515.

[11] J. B. T. M. Roerdink and A. Meijster, *The watershed transform: definitions, algorithms, and parallellization strategies*, Fundamenta Informaticae **41** (2000), 187-228. Report IWI 99-9-06, Institute for Mathematics and Computing Science, 1999.

[12] M. ten Caat, N. M. Maurits, and J. B. T. M. Roerdink, *Tiled Parallel Coordinates for the Visualization of Time-Varying Multichannel EEG data*, Proc. Eurographics/IEEE VGTC Symposium on Data Visualization (EuroVis), 2005, pp. 61–68.

[13] ———, *Design and Evaluation of Tiled Parallel Coordinate Visualization of Multichannel EEG Data*, IEEE Transactions on Visualization and Computer Graphics **13** (2007), no. 1, 70-79.

[14] ———, *Functional Unit Maps for Data-Driven Visualization of High-Density EEG Coherence*, Proc. Eurographics/IEEE VGTC Symposium on Data Visualization (EuroVis), 2007, pp. 259-266.

[15] E. Tomita, A. Tanaka, and H. Takahashi, *The worst-case time complexity for generating all maximal cliques and computational experiments*, Theor. Comput. Sci. **363** (2006), no. 1, 28–42.

[16] D. R. Wood, *An Algorithm for Finding a Maximum Clique in a Graph*, Operations Research Letters **21** (1997), no. 5, 211–217.

Watershed cuts

Jean Cousty, Gilles Bertrand, Laurent Najman and Michel Couprie

Université Paris-Est, LABINFO-IGM, UMR CNRS 8049, A2SI-ESIEE, France
{j.cousty,g.bertrand,l.najman,m.couprie}@esiee.fr

Abstract We study the watersheds in edge-weighted graphs. Contrarily to previous works, we define the watersheds following the intuitive idea of drops of water flowing on a topographic surface. We establish the consistency (with respect to characterizations of the catchment basins and dividing lines) of these watersheds, prove their optimality (in terms of minimum spanning forests) and derive a linear-time algorithm. To the best of our knowledge, similar properties are not verified in other frameworks and the proposed algorithm is the most efficient existing algorithm.

Keywords: watershed, catchment basin, minimum spanning forest.

Introduction

The watershed transform introduced by Beucher and Lantuéjoul [3] for image segmentation is used as a fundamental step in many powerful segmentation procedures. Many approaches [2, 6, 9, 14] have been proposed to define and/or compute the watershed of a vertex-weighted graph corresponding to a grayscale image. The digital image is seen as a topographic surface: the gray level of a pixel becomes the elevation of a point, the basins and valleys of the topographic surface correspond to dark areas, whereas the mountains and crest lines correspond to light areas.

In this paper, we investigate the watershed in a different framework: we consider a graph whose edges are weighted by a cost function (see, for example, [10] and [8]). A watershed of a topographic surface may be thought of as a separating line-set on which a drop of water can flow down toward several minima. To formalize this intuitive idea, we introduce the *drop of water principle* that leads to the definition of *watershed cuts* in edge-weighted graphs.
We establish the consistency of watershed cuts. In particular, we prove that they can be equivalently defined by their "catchment basins" (through a steepest descent property) or by the "dividing lines" separating these catchment basins (through the drop of water principle). As far as we know, in other frameworks, no similar property has ever been proved and some counter-examples showing that such a duality does not hold can be found.
We also establish the optimality of watershed cuts. In [10], F. Meyer shows the link between minimum spanning forests (MSF) and flooding from

marker algorithms. In this paper, we prove the equivalence between watershed cuts and separations induced by minimum spanning forests relative to the minima.
Finally, we propose a linear-time algorithm which does not require any sorting step, nor the use of any hierarchical queue, nor the extraction of the minima in a preprocessing step. This algorithm therefore runs in linear time whatever the range of the input map. To the best of our knowledge, this is the first watershed algorithm with such a property.

The proofs of the properties presented in this paper are given in an extended version [7].

1. Basic notions for edge-weighted graphs

We present some basic definitions to handle edge-weighted graphs.

We define a *graph* as a pair $X = (V(X), E(X))$ where $V(X)$ is a finite set and $E(X)$ is composed of unordered pairs of $V(X)$, i.e., $E(X)$ is a subset of $\{\{x, y\} \subseteq V(X) \mid x \neq y\}$. Each element of $V(X)$ is called a *vertex or a point (of X)*, and each element of $E(X)$ is called an *edge (of X)*. If $V(X) \neq \emptyset$, we say that X is *non-empty.*
Let X be a graph. If $u = \{x, y\}$ is an edge of X, we say that x and y are *adjacent (for X)*. Let $\pi = \langle x_0, \dots, x_l \rangle$ be an ordered sequence of vertices of X, π is *a path from x_0 to x_l in X (or in $V(X)$)* if for any $i \in [1, l]$, x_i is adjacent to x_{i-1}. In this case, we say that x_0 *and* x_l *are linked for* X. If $l = 0$, then π *is a trivial path in* X. We say that X *is connected* if any two vertices of X are linked for X.

Let X and Y be two graphs. If $V(Y) \subseteq V(X)$ and $E(Y) \subseteq E(X)$, we say that Y *is a subgraph of* X and we write $Y \subseteq X$. We say that Y is a *connected component of* X, or simply a *component of* X, if Y is a connected subgraph of X which is maximal for this property, i.e., for any connected graph Z, $Y \subseteq Z \subseteq X$ implies $Z = Y$.

Throughout this paper G denotes a connected graph. In order to simplify the notations, this graph will be denoted by $G = (V, E)$ instead of $G = (V(G), E(G))$. We will also assume that $E \neq \emptyset$.

For applications to image segmentation, we may assume that V is the set of picture elements (pixels) and E is any of the usual adjacency relations.

Let $X \subseteq G$. An edge $\{x, y\} \in E$ is *adjacent to* X if $\{x, y\} \cap V(X) \neq \emptyset$ and if $\{x, y\}$ does not belong to $E(X)$; in this case and if y does not belong to $V(X)$, we say that y is *adjacent to* X. If π is a path from x to y and y is a vertex of X, then π is a *path from x to X (in G).*

Let $S \subseteq E$. We denote by $\overline{S}$ *the complementary set of* S *in* E, i.e., $\overline{S} = E \setminus S$. *The graph induced by* S is the graph whose edge set is S and whose vertex set is made of all points which belong to an edge in S, i.e., $(\{x \in V \mid \exists u \in S, x \in u\}, S)$. In the following,the graph induced by S is also denoted by S.

We denote by $\mathcal{F}$ the set of all maps from E to $\mathbb{Z}$.

Let $F \in \mathcal{F}$. If u is an edge of G, $F(u)$ is the *altitude* of u. Let $X \subseteq G$ and $k \in \mathbb{Z}$. A subgraph X of G is a *(regional) minimum of F (at altitude k)* if: *i*) X is connected ; *ii*) k is the altitude of any edge of X; and *iii*) the altitude of any edge adjacent to X is strictly greater than k.
We denote by $M(F)$ the graph whose vertex set and edge set are, respectively, the union of the vertex sets and edge sets of all minima of F.

In the sequel of this paper, F denotes an element of $\mathcal{F}$.

For applications to image segmentation, we will assume that the altitude of u, an edge between two pixels x and y, represents the dissimilarity between x and y. Thus, we suppose that salient contours are located on the highest edges of G.

2. Watersheds

The intuitive idea underlying the notion of a watershed comes from the field of topography: a drop of water falling on a topographic surface follows a descending path and eventually reaches a minimum. The watershed may be thought of as the separating lines of the domain of attraction of drops of water. Despite its simplicity, none of the classical definitions handle this intuitive idea. In this paper, contrarily to previous works, we follow the drop of water principle to define a watershed in an edge-weighted graph.

Extensions and graph cuts

We present the notions of extension and graph cut which play an important role for defining a watershed in an edge-weighted graph.
Intuitively, the regions of a watershed (also called catchment basins) are associated with the regional minima of the map. Each catchment basin contains a unique regional minimum, and conversely, each regional minimum is included in a unique catchment basin: the regions of the watershed "extend" the minima. In [2], G. Bertrand formalizes the notion of extension.

Definition 1 (from Def. 12 in [2])**.** Let X and Y be two non-empty subgraphs of G. We say that Y is an *extension of X (in G)* if $X \subseteq Y$ and if any component of Y contains exactly one component of X. $\square$

The subgraphs in Figures 1(b) and 1(c) are two extensions of the one in Figure 1(a).

The notion of extension is very general. Many segmentation algorithms iteratively extend some seed components in a graph: they produce an extension of the seeds. Most of them terminate once they have reached an extension which cover all the vertices of the graph. The separation which is thus produced is called a graph cut.

Definition 2. Let $X \subseteq G$ and $S \subseteq E$. We say that S is a *(graph) cut for X* if $\overline{S}$ is an extension of X and if S is minimal for this property, i.e., if $T \subseteq S$ and $\overline{T}$ is an extension of X, then we have $T = S$. $\square$

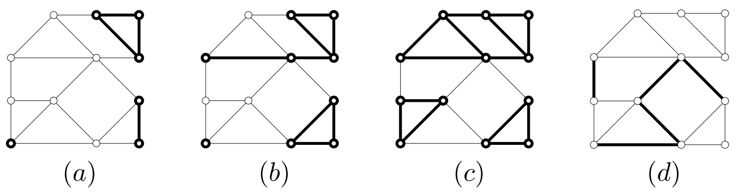


Figure 1. A graph G. The set of vertices and edges represented in bold is: (a), a subgraph X of G; (b), an extension of X; (c): an extension Y of X which is maximal; and (d): a cut S for X such that $\overline{S} = Y$.

The set S depicted in Figure 1(d) is a cut for X (Figure 1(a)). It can be verified that $\overline{S}$ (Figure 1(c)) is an extension of X and that S is minimal for this property.
If X is a subgraph of G and S a cut for X, it may be easily seen that $\overline{S}$ is a maximal extension of X.

The notion of graph cut has been studied for many years in graph theory. For applications to image segmentation, a classical problem is to find a cut of minimum weight (a min-cut) for a set of terminal points. The links between these approaches and the one developed in this paper are investigated in [1].

Watersheds by the drop of water principle

We introduce the watershed cuts of an edge-weighted graph. To this end, we formalize the drop of water principle. Intuitively, the catchment basins constitute an extension of the minima and they are separated by "lines" from which a drop of water can flow down towards distinct minima.

Let $\pi = \langle x_0, \ldots, x_l \rangle$ be a path in G. The path π is *descending (for F)* if, for any $i \in [1, l-1]$, $F(\{x_{i-1}, x_i\}) \geq F(\{x_i, x_{i+1}\})$.

Definition 3 (drop of water principle). Let $S \subseteq E$. We say that S *is a watershed cut (or simply a watershed) of* F if $\overline{S}$ is an extension of $M(F)$ and if for any $u = \{x_0, y_0\} \in S$, there exist $\pi_1 = \langle x_0, \ldots, x_n \rangle$ and $\pi_2 = \langle y_0, \ldots, y_m \rangle$ which are two descending paths in $\overline{S}$ such that:
- x_n and y_m are vertices of two distinct minima of F; and
- $F(u) \geq F(\{x_0, x_1\})$ (resp. $F(u) \geq F(\{y_0, y_1\})$), whenever π_1 (resp. π_2) is not trivial. □

In order to illustrate the previous definition, it may be seen that the set S of dashed edges in Figure 2(b) is a watershed of the corresponding map F. The minima of F are depicted in bold in Figure 2(a).

Let $S \subseteq E$. We remark that if S is a watershed of F, then S is necessarily a cut for $M(F)$. The converse is in general not true since a watershed of F is defined thanks to conditions that depend of the altitude of the edges whereas the definition of a cut is solely based on the structure of the graph.

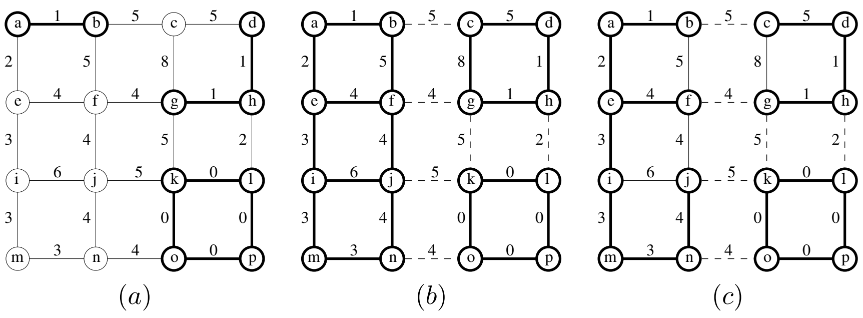


Figure 2. A graph G and a map F. Edges and vertices in bold depict: (a), the minima of F; (b), a maximal extension of $M(F)$; and (c), a MSF relative to $M(F)$. Dashed edges in (b) and (c) depict a watershed of F.

Catchment basins by a steepest descent property

A popular alternative to the drop of water principle defines a watershed exclusively by its catchment basins and does not involve any property of the divide.

In the framework of edge-weighted graph, we define a *catchment basin* as a component of the graph induced by the complementary set of a watershed.

The following theorem (Theorem 1) shows that a watershed can be defined equivalently by its divide line or by its catchment basins.

For that purpose, we start with some definitions relative to the notion of path with steepest descent.

From now on, we will also denote by F the map from V to $\mathbb{Z}$ such that for any $x \in V$, $F(x)$ is the minimal altitude of an edge which contains x, i.e., $F(x) = \min\{F(u) \mid u \in E, x \in u\}$; $F(x)$ is called the altitude *of x.*

Let $\pi = \langle x_0, \dots, x_l \rangle$ be a path in G. The path π is *a path with steepest descent for F* if, for any $i \in [1, l]$, $F(\{x_{i-1}, x_i\}) = F(x_{i-1})$.

Theorem 1 (consistency)**.** *Let $S \subseteq E$ be a cut for $M(F)$. The set S is a watershed of F if and only if there exists a path with steepest descent in the graph induced by $\overline{S}$ from each point of V to $M(F)$.*

The previous theorem establishes the consistency of watershed cuts: they can be equivalently defined by a steepest descent property on the catchment basins (regions) or by a the drop of water principle on the cut (frontier) which separates them. As far as we know, in the literature about discrete watersheds, no similar property ([11]) has ever been proved and some counter-examples showing that such a duality does not hold in other frameworks can be found. Hence, Theorem 1 emphasizes that edge-weighted graphs are adapted for the definition and study of watersheds.

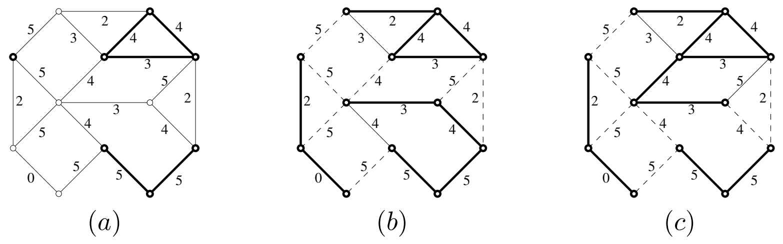


Figure 3. A graph G and a map F. The bold edges and vertices represent: (a), X a subgraph of G; (b) and (c), two MSFs relative to X; their induced cuts are represented by dashed edges.

3. Minimum spanning forests and watersheds

We introduce the minimum spanning forests relative to subgraphs of G and show the equivalence between the watershed cuts of a map and the cuts induced by minimum spanning forests relative to the minima of this map.

Let X and Y be two non-empty subgraphs of G. We say that Y is a *forest relative to* X if: *i)* Y is an extension of X; and *ii)* for any extension $Z \subseteq Y$ of X, we have $Z = Y$ whenever $V(Z) = V(Y)$.

We say that Y is a *spanning forest relative to* X *(for* G*)* if Y is a forest relative to X and $V(Y) = V$.

Thanks to the notion of relative forest, the usual notions of a tree and a forest can be defined as follows.

Let $X \subseteq G$. We say that X is a *tree* (resp. a *spanning tree*) if X is a forest (resp. spanning forest) relative to the subgraph $(\{x\}, \emptyset)$, x being any vertex of X. We say that X is a *forest* (resp. a *spanning forest*) if X is a forest (resp. a spanning forest) relative to $(S, \emptyset)$, S being a subset of $V(X)$.

Let $X \subseteq G$, *the weight of* X *(for* F*)* is the value $F(X) = \sum_{u \in E(X)} F(u)$.

Definition 4. Let X and Y be two subgraphs of G. We say that Y *is a minimum spanning forest (MSF) relative to* X *(for* F*, in* G*)* if Y is a spanning forest relative to X and if the weight of Y is less than or equal to the weight of any other spanning forest relative to X. □

For instance, the graphs Y and Z (bold edges and vertices) in Figures 3(b) and 3(c) are two MSFs relative to X (Figure 3(a)).

We now have the mathematical tools to present the main result of this section (Theorem 2) which establishes the optimality of watersheds.

Let X be a subgraph of G and let Y be a spanning forest relative to X. There exists a unique cut for Y and this cut is also a cut for X. We say that this unique cut is the *cut induced by* Y.

Theorem 2 (optimality)**.** *Let* $S \subseteq E$*. The set* S *is a cut induced by a MSF relative to* $M(F)$ *if and only if* S *is a watershed cut of* F*.*

A MSF relative to the minima of the map Figure 2 is depicted in bold in Figure 2(c). Observe that the induced cut is indeed a watershed.

As far as we know, this is the first result which establishes watershed optimality in graphs. Furthermore, Theorem 2 suggests that MSF is a method of choice for marker based watershed, an illustration of which is given in [7].

The minimum spanning tree problem is one of the most typical and well-known problems of combinatorial optimization (see [5]). In the next paragraph, we show that the minimum spanning tree problem is equivalent to the problem of finding a MSF relative to a subgraph of G.

Let $X \subseteq G$. The graph *X is a minimum spanning tree (for F, in G)* if X is a MSF relative to the subgraph $(\{x\}, \emptyset)$, x being any vertex of X.

In order to recover the link between flooding algorithms and minimum spanning tree algorithms, in [10], F. Meyer proposed a construction which shows the equivalence between finding a MSF rooted in a set of vertices (*i.e.*, a MSF relative to a subgraph X of G such that $E(X) = \emptyset$) and finding a minimum spanning tree. This construction can be easily extended for proving the equivalence between finding a minimum spanning tree and a MSF relative to any subgraph X of G. To this end, in a preliminary step, each component of X must be contracted into a single vertex and, if two vertices of the contracted graph are linked by multiple edges only the one with minimal value is kept.

A direct consequence is that any minimum spanning tree algorithm can be used to compute a relative MSF. Many efficient algorithms (see a survey in [5]) exist in the literature for solving the minimum spanning tree problem.

4. Streams and linear-time watershed algorithm

As seen in the previous section, MSFs relative to subgraphs of G, and by the way watershed cuts, can be computed by any minimum spanning tree algorithm. The best complexity for solving this problem is reached by the quasi-linear algorithm of Chazelle [4]. In this section, we introduce a linear-time watershed algorithm. This algorithm does not require any sorting step nor the use of any hierarchical queue. Thus, whatever the range of the considered map, it runs in linear time with respect to the size of the input graph. Furthermore, this algorithm does not need to compute the minima of the map in a preliminary step. To the best of our knowledge, this is the first watershed algorithm with such properties.

We first introduce the mathematical tools which allow us to prove the correctness of the proposed algorithm. In particular, we propose a notion of stream which is crucial to this paradigm. Then, the algorithm is presented, and both its correctness and complexity are analyzed.

Definition 5. Let $L \subseteq V$. We say that L is a *stream* if, for any two points x and y of L, there exists, in L, either a path from x to y or from y to x, with steepest descent for F. Let L be a stream and let $x \in L$. We say that *x is a top (resp. bottom) of L* if the altitude of x is greater than (resp. less than) or equal to the altitude of any $y \in L$. $\square$

We remark that if L is a stream and x is a bottom (resp. a top) of L, then, from any $y \in L$ to x (resp. from x to any $y \in L$), there is a path in L, with steepest descent for F. Notice that, whatever the stream L, there exists a top (resp. bottom) of L. Nevertheless, this top (resp. bottom) is not necessarily unique.

In order to illustrate the previous definitions, let us assume that G and F are the graph and the function depicted in Figure 2. The sets $L = \{a, b, e, i\}$ and $\{j, m, n\}$ are two examples of streams. On the contrary, the set $L' = \{i, j, k\}$ is not a stream since there is no path in L', between i and k, with steepest descent for F. The sets $\{a, b\}$ and $\{i\}$ are respectively the set of bottoms and tops of L.

The algorithm which will be proposed in this section is based on the iterative extraction of streams. In order to build such a procedure, we study stream concatenation.

Let L_1 and L_2 be two disjoint streams (i.e., $L_1 \cap L_2 = \emptyset$) and let $L = L_1 \cup L_2$. We say that *L_1 is under L_2*, written $L_1 \prec L_2$, if there exist a top x of L_1, a bottom y of L_2, and there is, from y to x, a path in L with steepest descent for F. Note that, if $L_1 \prec L_2$, then L is also a stream.
We say that a stream L is an *$\prec$-stream* if there is no stream under L.

In Figure 2(a) the stream $\{a, b, e, i\}$ is under the stream $\{j, m, n\}$ and thus $\{a, b, e, i, j, m, n\}$ is also a stream. Furthermore, there is no stream under $\{a, b, e, i\}$ and $\{a, b, e, i, j, m, n\}$. Thus, these are two $\prec$-streams.

The streams extracted by our algorithm are $\prec$-streams. As said in the introduction, this algorithm does not require minima precomputation. In fact, there is a deep link between $\prec$-streams and minima.

Proposition 1. *Let L be a stream. The three following statements are equivalent:*
(1) *L is an $\prec$-stream;*
(2) *L contains the vertex set of a minimum of F; and*
(3) *for any $y \in V \setminus L$ adjacent to a bottom x of L, $F(\{x, y\}) > F(x)$.*

In Figure 2(a) the two $\prec$-streams $\{a, b, e, i\}$ and $\{a, b, e, i, j, m, n\}$ contain the set $\{a, b\}$ which is the vertex set of a minimum of F.

In order to partition the vertex set of G, from the $\prec$-streams of F, the vertices of the graph can be arranged in the following manner.

Let $\mathcal{L} = \{L_1, \ldots, L_n\}$ be a set of n $\prec$-streams. We say that $\mathcal{L}$ is a *flow family* if: *i)* $\cup\{L_i \mid i \in \{1, \ldots, n\}\} = V$; and *ii)* for any two distinct L and L' in $\mathcal{L}$, if $L \cap L' \neq \emptyset$, then $L \cap L'$ is the vertex set of a minimum of F.

Let $\mathcal{L}$ be a flow family and let $x \in V$. It may be seen that, either x belongs to a minimum of F (in this case, it may belong to several elements of $\mathcal{L}$), or x belongs to a unique $\prec$-stream of $\mathcal{L}$ which itself contains the vertex set of a unique minimum of F. Thus, thanks to $\mathcal{L}$, we can associate to each vertex x of G a unique minimum of F.

Definition 6. Let $\mathcal{L}$ be a flow family. Let us denote by $M_1, \ldots, M_n$ the minima of F. Let ψ be the map from V to $\{1, \ldots, n\}$ which associates to each vertex x of V, the index (or label) i such that M_i is the unique minimum of F included in an $\prec$-stream of $\mathcal{L}$ which contains x; we say that ψ is a *flow mapping of* F. If ψ is a flow mapping of F, we say that the set $S = \{\{x, y\} \in E \mid \psi(x) \neq \psi(y)\}$ is a *flow cut for* F. □

The next proposed algorithm produces a flow mapping, and hence a flow cut. The following theorem, which states the equivalence between flow cuts and watersheds, is the main tool to establish the correctness of Algorithm 1.

Theorem 3. *Let $S \subseteq E$. The set S is a watershed of F if and only if S is a flow cut for F.*

We now present Algorithm 1 which computes a flow mapping, hence, by Theorem 3, a watershed. It iteratively assigns a label to each point of the graph. To this end, from each non-labeled point x, a stream L composed of non-labeled points and whose top is x is computed (Line 4). If L is an $\prec$-stream (Line 5), a new label is assigned to the points of L. Otherwise (Line 8), there exists an $\prec$-stream L' under L and which is already labeled. In this case, the points of L receive the label of L' (Line 9). The function `Stream`, called at Line 4, computes the stream L. Roughly speaking, it performs an intermixed sequence of depth-first and breadth-first exploration of the paths with steepest descent. The main invariants of the function `Stream` are: *i*), the set L is, at each iteration, a stream; and *ii*), the set L' is made of all non-already explored bottoms of L. The function halts at Line 17 when all bottoms of L have been explored or, at Line 9, if a point z already labeled is found. In the former case, by Proposition 1, the returned set L is an $\prec$-stream. In the latter case, the label *lab* of z is also returned and there exists a bottom y of L such that $\langle y, z \rangle$ is a path with steepest descent. Thus, there is an $\prec$-stream L', under L, included in the set of all vertices labeled *lab*. Thus, by the preceding remarks, the output of Algorithm 1 is a flow mapping of F. Furthermore, using classical data structures to represent the graph G, we obtain a linear complexity.

Proposition 2. *Algorithm 1 outputs a map ψ which is a flow mapping of F. Furthermore, Algorithm 1 runs in linear-time with respect to $|E|$.*

5. Illustration

In order to illustrate the use of watershed cuts in practical applications, we derive, from the classical framework of mathematical morphology, a segmentation scheme which consists in the three following steps: (*i*), computation

Algorithm 1: Flow mapping.

```
Data: $(V, E, F)$: an edge-weighted graph;
Result: $\psi$: a flow mapping of $F$.
1 foreach $x \in V$ do $\psi(x) := NO\_LABEL$;
2 $nb\_labs := 0$ ; /* the number of minima already found */
3 foreach $x \in V$ such that $\psi(x) = NO\_LABEL$ do
4     $[L, lab] :=$ Stream$(V, E, F, \psi, x)$ ;
5     if $lab = -1$ then /* $L$ is an $\prec$-stream */
6         $nb\_labs + +$ ;
7         foreach $y \in L$ do $\psi(y) := nb\_labs$;
8     else
9         foreach $y \in L$ do $\psi(y) := lab$;
```

of a simple function that assigns a weight to the edges of the 4-adjacency graph associated to the image; (*ii*), filtering of this cost function in order to reduce the number of minima; and (*iii*) computation of a watershed of the filtered cost function.

We assume that the graph G is the one corresponding to the 4-adjacency relation associated to the image I which is to be segmented. We consider also the map F defined for any $\{x, y\} \in E$ by $F(\{x, y\}) = |I(x) - I(y)|$.

A watershed of F would contain too many catchment basins. In order to suppress many of the non-significant minima, a classical approach consists of computing morphological filtering of the function [13]. For this illustration, we implement an adaptation of a classical filter which consists of: (*i*) remove the connected component of a lower threshold of F with minimal area; (*ii*), repeat step (*i*) until F has k (a predefined number) minima. Hence, the watershed of the filtered map contains exactly k catchment basins. The results obtained on the cameraman image ($k = 22$) are presented in Figure 4.

Conclusion and perspectives

In this paper, we introduce watershed cuts, a notion of watershed in edge-weighted graphs. We show the consistency and optimality of watershed cuts. Furthermore, we derive a simple algorithm which runs in linear-time whatever the range of the input map. For more details on watershed cuts, we refer to [7]. In particular in [7], we show that a watershed cut is a separation which corresponds to a separation produced by a topological watershed [2, 6] defined on edge-weighted graphs. We also study the links with shortest-path forests [8].

Further work will be focused on hierarchical segmentation schemes based on watershed cuts (including *geodesic saliency of watershed contours* [12]

Function `Stream(` V, E, F, ψ, x`).`

```
Data: $(V, E, F)$: an edge-weighted graph; $\psi$: a labeling of $V$; $x$:
      a point in $V$.
Result: $[L, lab]$ where $L$ is a stream such that $x$ is a top of $L$,
        and $lab$ is either a label of an $\prec$-stream under $L$ or $-1$.
1  $L := \{x\}$ ;
2  $L' := \{x\}$ ; /* the set of non-explored bottoms of $L$ */
3  while there exists $y \in L'$ do
4      $L' := L' \setminus \{y\}$;
5      $breadth\_first := TRUE$ ;
6      while $(breadth\_first)$ and (there exists $\{y, z\} \in E$ such
       that $z \notin L$ and $F(\{y, z\}) = F(y)$) do
7          if $\psi(z) \neq NO\_LABEL$ then
8              /* there is an $\prec$-stream under $L$ already labelled */
9              return $[L, \psi(z)]$ ;
10         else if $F(z) < F(y)$ then
11             $L := L \cup \{z\}$ ; /* $z$ is now the only bottom of $L$ */
12             $L' := \{z\}$ ; /* hence, switch to depth-first exploration */
13             $breadth\_first := FALSE$ ;
14         else
15             $L := L \cup \{z\}$ ; /* $F(z) = F(y)$, thus $z$ is also a bottom
               of $L$ */
16             $L' := L' \cup \{z\}$ ; /* continue breadth-first exploration */
17 return $[L, -1]$ ;
```

and *incremental MSFs*) as well as on watershed in weighted simplicial complexes, an image representation adapted to the study of topological properties. Furthermore, we intend to show that our watershed algorithm can be used to efficiently compute minimum spanning trees.

References

[1] C. Allène, J. Y. Audibert, M. Couprie, J. Cousty, and R. Keriven, *Some links between min cuts, optimal spanning forests and watersheds*, Procs. 8th ISMM, 2007. These proceedings.

[2] G. Bertrand, *On topological watersheds*, JMIV **22** (2005), no. 2-3, 217–230.

[3] S. Beucher and C. Lantuéjoul, *Use of watersheds in contour detection*, Procs. of the international workshop on image processing real-time edge and motion detection/estimation, 1979.

[4] B. Chazelle, *A minimum spanning tree algorithm with inverse-Ackermann type complexity*, Journal of the ACM **47** (2000), 1028–1047.

[5] T. H. Cormen, C. Leiserson, and R. Rivest, *Introduction to algorithms, second edition*, MIT Press, 2001.

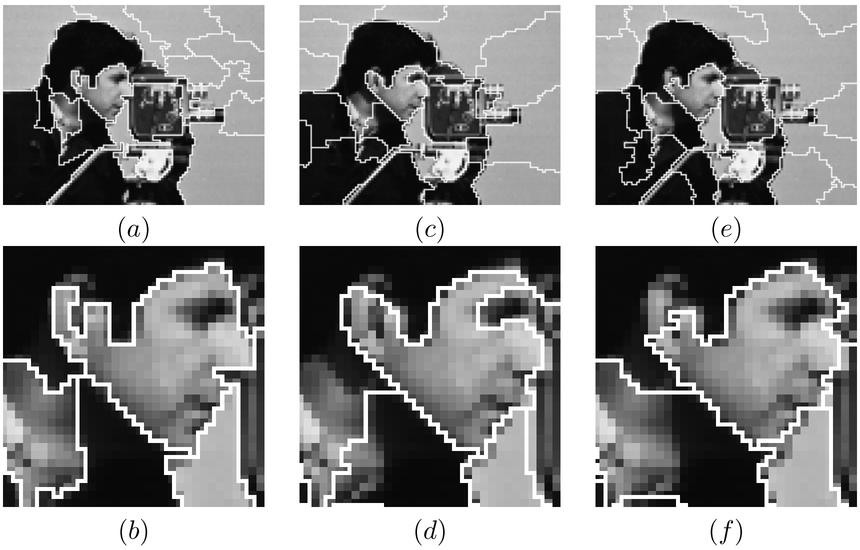


Figure 4. Results obtained by applying a grayscale watershed on a filtered map [see text]. (a, b) A watershed cut superimposed in white to the original image I; (c, d) a watershed by flooding of the filtered Deriche optimal edge detector; (e, f) a watershed by flooding of a filtered morphological gradient. In the three cases, a similar procedure based on area filtering is used to ensure that the watershed has exactly 22 catchment basins.

[6] M. Couprie and G. Bertrand, *Topological grayscale watershed transform*, Procs. of spie vision geometry v, 1997, pp. 136–146.

[7] J. Cousty, G. Bertrand, L. Najman, and M. Couprie, *Watersheds, minimum spanning forests and the drop of water principle*, Technical report IGM2007-01 (2007).

[8] A. X. Falcão, J. Stolfi, and R. de Alencar Lotufo, *The image foresting transform: theory, algorithm and applications*, IEEE Trans. PAMI **26** (2004), 19–29.

[9] F. Meyer, *Un algorithme optimal de ligne de partage des eaux*, Procs. of 8ème congrès afcet, 1991, pp. 847–859.

[10] ———, *Minimum spanning forests for morphological segmentation*, Procs. of the second international conference on mathematical morphology and its applications to image processing, 1994, pp. 77–84.

[11] L. Najman and M. Schmitt, *Watershed of a continuous function*, Signal Processing **38** (1993), no. 1, 68–86.

[12] ———, *Geodesic saliency of watershed contours and hierarchical segmentation*, IEEE Trans. PAMI **18** (1996), no. 12, 1163–1173.

[13] J. Serra and L. Vincent, *An overview of morphological filtering*, Circuits Systems Signal Process **11** (1992), no. 1, 48–107.

[14] L. Vincent and P. Soille, *Watersheds in digital spaces: An efficient algorithm based on immersion simulations*, IEEE Trans. PAMI **13** (1991), no. 6, 583–598.

Watershed from propagated markers improved by a marker binding heuristic

FRANKLIN C. FLORES[1,2] and ROBERTO DE A. LOTUFO[2]

[1] *Departamento de Informática, Universidade Estadual de Maringá, Brazil*
`fcflores@din.uem.br`

[2] *Faculdade de Engenharia Elétrica e de Computação (FEEC), Universidade Estadual de Campinas (Unicamp), SP, Brazil*
`lotufo@unicamp.br`

Abstract Watershed from propagated markers is a generic method to interactive segmentation of objects in image sequences. It consists in a combination of classical watershed from markers with motion estimation techniques. In order to improve the watershed from propagated markers technique, this paper introduces a marker binding heuristic. It consists in the imposition of pairs of markers along the border of the object of interest and both markers in a pair, the internal and external ones, must be propagated by the same displacement vector computed from the regions delimited by the pair.

Keywords: watershed, propagated markers, object segmentation, image sequences, binding of markers.

1. Introduction

Object segmentation in image sequences [6,7,13] is the segmentation frame-to-frame of an object which semantics remains unchanged. Such technique have been successfully applied to video edition (*Video Masking*) [5,6,11,13, 15,21], video coding [17,19,22,23,25], video surveillance [9,19] and biomedical imaging [1,20].

There are two categories of techniques to segment objects in image sequences: *automatic* (or *non-supervised*) and *assisted* (or *supervised* or also *interactive*). In the automatic segmentation, the objects are detected automatically in the initial frame and they are tracked in the following frames, through application of motion estimation techniques. In the automatic segmentation there is no intervention by users in the obtained results.

In the assisted segmentation, the user is allowed to intervene in the segmentation process. The user can choose, for instance, the objects to be segmented, how they will be tracked, and he/she has the option to correct and alter the segmentation results.

It was proposed, in a previous paper, the *watershed from propagated markers*, a generic method to interactive segmentation of objects in image

G. Banon, et al. (eds),
Mathematical Morphology and its Applications to Signal and Image Processing, 313–323.
©2007 MCT/INPE. Printed in Brazil.
URL of this article: http://urlib.net/dpi.inpe.br/ismm@80/2007/03.03.03.06

sequences [7]. This method consists in a combination of the watershed from markers [4, 24] with motion estimation techniques [2, 3]. The segmentation technique is tied to the motion estimation one, since the markers to the objects of interest are propagated to the next frames in order to track such objects.

The watershed from propagated markers presents the following characteristics.

1. Interactivity: the user may intervene in the segmentation results: it must be allowed to the user to add/remove markers, to correct bad segmentation and to choose how the markers will be propagated.

2. Generality: the technique can be applied to any image sequences. It is not necessary any *a priori* knowledge about the sequence.

3. Rapid Response: once a marker is imposed or the propagation is activated, the method must answer quickly.

4. Progressive Manual Edition: the user does not need to "look back" to check the previous segmentation; they are considered done. It is not also necessary to erase all markers imposed to a frame when a bad segmentation occurs; the bad segmentation is locally fixed by adding/removing markers to this region.

The proposed method consists in the imposition of markers to the objects of interest in the current frame, given their computation from the segmentation results in the previous frame and their propagation to the current frame, in order to adjust them to their respective objects.

Each marker is propagated from the previous frame to the current one by a displacement vector given by the motion estimation in the area where the marker was computed.

The computation of the internal (external) markers to an object is done by taking the contour of the erosion (dilation) of the object segmentation result in the previous frame. This contour is broken in short segments, and each segment is a marker belonging to the set of internal (external) markers.

A reasonable assumption about the marker propagation is that two closer markers assigned to the same object should have similar displacement vectors, i.e., both markers should follow the motion of the object. However, there are situations where this does not occur properly for two reasons:

1. a marker, that consists in a short segment, may not provide enough information to estimate accurately its motion;

2. the motion of two closer markers are computed separetely.

In these cases, the motion of these markers may not be coherent, or even wrong.

This paper presents an improvement to the watershed from propagated markers: *the binding of markers*. It consists of computing pairs of markers along the border, and each pair is composed by an internal marker and an external one. Both markers in the pair must be propagated by the same displacement vector, and this vector is computed by the motion estimation of the area *between* the pair of markers.

The binding of markers provides more information to the motion estimation (both markers *and* the area between them). More, it helps the motion of the pair of markers to follow the motion of the border that crosses the region between them in the previous frame.

This paper is organized as follows: The watershed from propagated markers is proposed and discussed in Section 2. The marker binding heuristic is introduced in Section 3. Several experimental results are presented and discussed in Section 4. Finally, this paper is conclude in Section 5.

2. Watershed from propagated markers

The watershed from propagated markers [7, 8] consists, basically, in the following steps.

1. The objects of interest are segmented by the interactive watershed from markers [18], in the initial frame.

2. Given the mask of the segmented objects, the **contour** of erosion of the object and the **contour** of the erosion of the background are obtained. Both contours are broken in short segments forming the set of inner and the set of outer markers to each object.

3. Each segment is propagated to the next frame by motion estimation [10, 14, 16]. These new set of inner and outer markers are used to apply the watershed technique to the next frame.

4. If necessary, the user interacts with the markers, doing the corrections by adding or removing markers, in order to fix the segmentation result.

5. Go to Step 2, until all sequence is processed.

The method proposed above works fine with bad defined contours or strongly textured objects, since the markers are imposed close to the borders of the objects to be segmented. If the quality of segmentation is not approved in some frame, the user can easily move the short-segment marker. The marker propagation is very fast since each segment consists in a few points. Moreover, the contours follow the object deformation, since new markers are created from the segmentation of each frame. The object to be segmented is processed until the end of sequence or until it leaves the scene or be totally occluded. If is partially occluded, it may be possible that the user should intervene to regularize the process.

3. The binding of markers

Let us consider a pair of closer markers (one internal and the other external) assigned to the same object. The heuristic proposed here is based on two assumptions:

1. the border of the object to be segmented crosses the region delimited by both markers of the pair (Figure 1(a));

2. the pair of markers must follow the motion of the border.

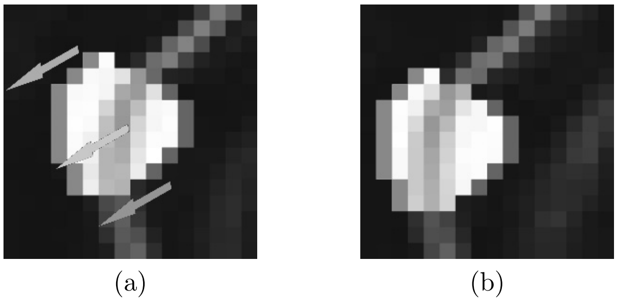

(a) (b)

Figure 1. Binding of Markers. (a) The internal (right) and external (left) markers delimite a region crossed by the object border. (b) Markers are propagated by the same displacement vector, computing in function of the delimited region.

Considering the above assumptions, both markers of the pair must be propagated by the same displacement vector, i.e., they are propagated to the same direction. For purpose of computing such displacement vector, the region delimited by both markers is considered as a "marker" (see Figure 1(a) - such region is the negated one located between the markers); the displacement vector computed to this region is assigned to the pair of markers. Since it is expected that a region located at the border of the object of interest in frame $k + 1$ gives the best match to the region delimited by the bound markers (Figure 1(b)), such markers should track the border of the object.

Figure 1 illustrates the idea. It shows the morphological gradient of two consecutive frames (both gradients are zoomed). Figure 1(a) shows a pair of markers, an internal (right) and an external (left). The area delimited by the markers was highlighted by negating the gradient at the area. The motion estimation is done considering that area as the marker to be propagated. The displacement vector assigned to this area (as illustrated in Figure 1(a) by the central vector) must be used as the displacement vector of the markers bound to this area (the right and left vectors are exemplified in Figure 1(a)). Figure 2(b) shows the pair of markers propagated to the next frame.

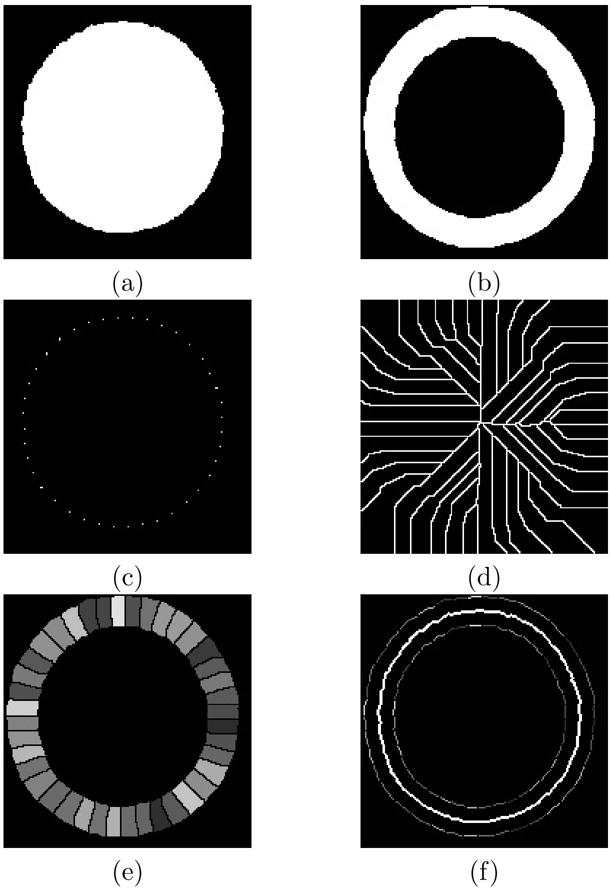


Figure 2. Creation of the marker pairs and the regions delimited by them. (a) Mask of the segmented object. (b) The "crown" of the mask that must be sliced in order to obtain the delimited regions. (c) The seeds to be used to slice the "crown" of the mask. (d) Watershed lines. (e) Delimited regions (labeled). (f) Pair of markers (labeled) wrapping the borders of the mask.

The pair of markers assigned to the object of interest and the region used to estimate the displacement vector are created by morphological processing of the mask M of the segmented object (Figure 2(a)). Two parameters are required: the distance $\mathbf{m}$ between the internal and the external markers and the width $\mathbf{w}$ of the area.

Let $M_{\delta,\mathbf{m}}$ and $M_{\varepsilon,\mathbf{m}}$ be, respectively, the dilation and the erosion of the mask M by a disk structuring element with diameter $\mathbf{m}$. Both images will be used to compute a "crown", by subtracting $M_{\varepsilon,\mathbf{m}}$ from $M_{\delta,\mathbf{m}}$ (Figure 2(b)).

The next step consists of creating seeds that will be used to separate the regions. Such seeds must be imposed on the contour of M and the geodesic distance [12] (started from a point picked from the contour) between them must be $\mathbf{w}$. It is done by labeling the contour of M using the geodesic distance function and by analyzing the division of each label by $\mathbf{w}$; the points which label divided by $\mathbf{w}$ remains zero are the seeds (Figure 2(c)).

The seeds will be used as markers in an application of the watershed operator. The resulting watershed lines (Figure 2(d)) will be used to slice the "crown" of the mask.

The slicing is done by computing the intersection between the negation of the watershed lines and the "crown". Figure 2(e) shows the sliced regions identified by a label (let L be the image which contains the labeled regions).

Given the set of all delimited areas, the creation of the markers is simple. To create the internal markers, just compute the intersection between L and the contour of $M_{\varepsilon,\mathbf{m}}$. To create the external markers, compute the intersection between L and the contour of $M_{\delta,\mathbf{m}}$. Figure 2(f) shows the pair of markers wrapping the borders of the mask M. Each pair of internal and external received a distinct label.

4. Experimental results

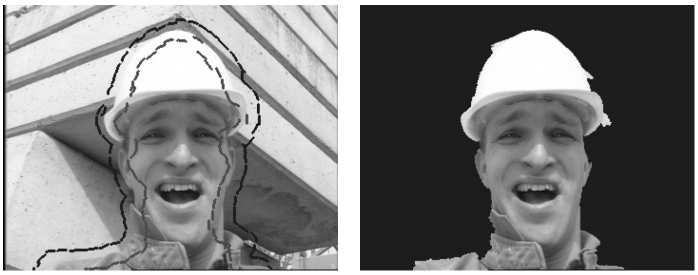
This section presents some experiments done with the watershed from propagated markers with the marker binding heuristic and their respective results. The first experiment demonstrates the improvement given by the binding of markers. The second one quantifies the method robustness with several test cases.

4.1 Binding of markers versus no heuristic

The goal in this experiment was to evaluate the improvement of the watershed from propagated markers by application of the marker binding heuristic. Figure 3 shows the propagation and segmentation results achieve by the heuristic.

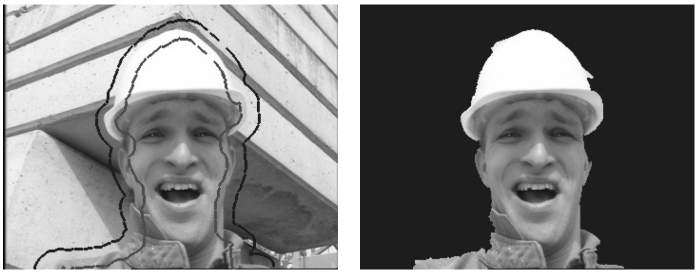
Figure 3(a) shows the marker propagation by Lucas-Kanade estimation, without adjustment. The length of each marker is $\mathbf{m} = 10$ pixels and the distance of each marker and the border (before propagation) is $\mathbf{w} = 10$ pixels. The result is good except for a few misplaced marker that led to a bad segmentation in some regions (Figure 3(b)).

The heuristic of bind pair of markers provided best results (Figure 4(a)). The fact that both inner and outer markers of the pair were propagated by the same displacement vector avoids the local crossing of the markers (i.e., the internal marker of the pair is not propagated outer than the external one, and vice-versa). More, despite the region between the markers is greater than the markers themselves, it provides more information than the markers without significant loss of performance. The segmentation errors occurred in Figure 4(b) is due the segmentation itself and not due to marker propagation.



(a) (b)

Figure 3. Heuristic comparison applied to the *Foreman* sequence. (a) No heuristic. (b) Segmentation result.



(a) (b)

Figure 4. Heuristic comparison applied to the *Foreman* sequence. (a) Binding of markers. (b) Segmentation result.

4.2 Robustness

It were done two experiments in order to assess the robustness of the watershed from propagated markers (using the Lucas-Kanade marker propagation and the heuristic of bind of markers).

The results of both experiments were compared to the result of sequence segmented and tracked *manually*. The robustness was assessed by computing, to each frame, the symmetrical difference between the manual segmentation and the segmentation provided by the application of the proposed method in the experiment. The percentage of pixels in the frame that is not zero (i.e., that belongs to the symmetrical difference) is the percentage of segmentation error to this frame.

In the first experiment, the object was segmented and tracked *without* user intervention. The user just insert markers to the first frame and call for propagation until the end of sequence, without marker edition. After the sequence is entirely segmented, the percentage of segmentation error was computed to each frame.

The second experiment consisted of applying the following instructions to each frame:

1. the percentage of segmentation error for this frame is computed, given the segmentation provided by the markers propagated from the previous frame;

2. the user *intervenes* and edit the markers, in order to correct the segmentation errors in the current frame;

3. the new segmentation of the current frame will provide the markers to be propagated to the next frame.

This experiment was done in order to illustrate the reduction in the percentage of segmentation error, when the user intervenes to correct the segmentation results.

It were segmented and tracked objects in several classical image sequences, and both experiments were done to each sequence. Table 1 shows the percentage of segmentation error in frames 1 to 8 to each sequence. The lines which sequence names are not bold contain the percentage of segmentation error when the method is applied without user intervention (first experiment). The other lines which sequence names are bold show the percentages of segmentation error when the user intervenes (second experiment).

Note that the error in the first frame to all experiments is zero, because, since the object of interest in the first frame is segmented manually, its segmentation result is equal to the segmentation of the same object in the sequence segmented manually. The percentages of segmentation error are the same in the second frame, to each sequence, because the markers provided to the second frame, in both experiments, come from a frame segmented manually. Finally, note the error reduction in each frame, provided by the user intervention in the current segmentation results.

5. Conclusion

In a previous paper, it was proposed the watershed from propagated markers, a generic method to interactive segmentation of objects in image sequences. It consists of computing short segments close to the object borders and apply them as markers propagated to the next frame. The marker propagation is done by motion estimation techniques and the segmentation of the obbjects of interest is done by classical watershed from markers technique. Besides the interactive and the generality, this method also presents two other main characteristics: progressive manual edition and rapid response.

Despite it is expected that two closer markers are propagated by similar displacement vectors, it sometimes does not occur, since the computation of

Table 1. Robustness: Percentage of segmentation error.

Sequence	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
Akiyo	0.00	0.85	1.09	0.82	0.95	0.78	0.70	1.20
Akiyo	0.00	0.85	0.95	0.74	0.91	0.76	0.67	0.91
Bream	0.00	0.80	1.72	1.85	0.00	1.98	1.92	1.99
Bream	0.00	0.80	0.61	0.77	1.25	0.59	0.25	0.46
Carphone	0.00	0.66	1.07	1.38	1.90	1.72	1.67	1.67
Carphone	0.00	0.66	0.92	1.31	1.34	1.04	1.38	1.63
Children	0.00	1.14	1.75	2.43	3.18	3.90	5.04	5.19
Children	0.00	1.14	1.31	1.57	1.57	1.93	1.59	2.18
Foreman	0.00	1.31	2.14	2.09	2.39	3.08	3.55	3.83
Foreman	0.00	1.31	2.09	1.15	1.83	1.24	0.98	0.50
Weather	0.00	1.49	1.70	2.36	2.31	2.17	2.26	2.33
Weather	0.00	1.49	1.44	1.52	1.50	1.34	1.28	1.44

the displacement vectors applied to each of such markers is done separetely or the information provided to the motion estimators is not sufficient to estimate accurately the marker motions.

This paper introduces the *binding of markers*, an heuristic applied to improve the watershed from propagated markers technique. It consists in the imposition of pairs of markers along the border of the object of interest, and both markers of each pair, an internal and an external ones, must be propagated by the same displacement vector, computed in function of the regions located between the two markers in the pair.

The contributions of the marker binding heuristic to the watershed from propagated markers are:

- the increasing in the amount of information provided to the motion estimation, which gives more accurate displacement vectors;
- the easiness for the pair of markers to follow the motion of the border that crosses the region between them in the previous frame.

Several experiments were also done in order to test the watershed from propagated markers with the marker binding heuristic. In comparison to the watershed from propagated markers as it was firstly proposed, the addition of the marker binding heuristic provided better results. It also worked fine when applied to a noisy sequence. Percentages of segmentation error were computed in the robustness experiment and its errors were low.

Future works include the design of more heuristics to boost the marker propagation and the segmentation results. One of this heuristics consists of correcting locally the segmentation result by tightening the pair of markers to the local border of the object.

References

[1] N. Ayache, P. Cinquin, I. Cohen, L. Cohen, F. Leitner, and O. Monga, *Segmentation of Complex Three-Dimensional Medical Objects: A Challenge and a Requirement for Computer-Assisted Surgery Planning and Performance* (1996), 59–74.

[2] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, *Performance ofOptical Flow Techniques*, International Journal of Computer Vision **12** (1994), no. 1, 43–77.

[3] S. S. Beauchemin and J. L. Barron, *The Computation of Optical Flow*, ACM Computing Surveys **27** (1995), no. 3, 433–467.

[4] S. Beucher and F. Meyer, *Mathematical Morphology in Image Processing* (1992), 433–481.

[5] C. Finch, *Special Effects: Creating Movie Magic*, Abbeville Press, 1984.

[6] F. C. Flores, R. Hirata Jr., J. Barrera, R. A. Lotufo, and F. Meyer, *Morphological Operators for Segmentation of Color Sequences*, IEEE Proceedings of SIBGRAPI'2000, 2000, pp. 300–307.

[7] F. C. Flores and R. A. Lotufo, *Object Segmentation in Image Sequences by Watershed from Markers: A Generic Approach*, IEEE Proceedings of SIBGRAPI'2003, 2003, pp. 347–352.

[8] F. C. Flores and R. A. Lotufo, *Watershed from Propagated Markers: An Interactive Method to Morphological Object Segmentation in Image Sequences*, submitted for publication.

[9] S. Gil, R. Milanese, and T. Pun, *Comparing Features for Target Tracking in Traffic Scenes*, Pattern Recognition **29** (1996), no. 8, 1285–1296.

[10] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison-Wesley Publishing Company, 1992.

[11] C. Gu and M.-C. Lee, *Semiautomatic Segmentation and Tracking of Semantic Video Objects*, IEEE Transactions on Circuits and Systems for Video Technology **8** (1998), no. 5, 572–584.

[12] H. J. A. M. Heijmans, *Morphological Image Operators*, Academic Press, Boston, 1994.

[13] R. Hirata Jr., J. Barrera, F. C. Flores, and R. A. Lotufo, *Automatic Design of Morphological Operators for Motion Segmentation*, IEEE Proceedings of Sibgrapi'1999, 1999, pp. 283–292.

[14] B. K. P. Horn and B.G. Schunck, *Determining Optical Flow*, Artificial Intelligence **17** (1981), 185–204.

[15] Na Li, Shipeng Li, Wen-Yin Liu, and Chun Chen, *A novel framework for semi-automatic video object segmentation*, IEEE International Symposium on Circuits and Systems, 2002 (ISCAS), 2002, pp. 811–814.

[16] B. Lucas and T. Kanade, *An interative image registration technique with an application to stereo system*, Proceedings of DARPA Image Understanding Workshop, 1981, pp. 121–130.

[17] M. Maziere, F. Chassaing, L. Garrido, and P. Salembier, *Segmentation and tracking of video objects for a content-based video indexing context*, IEEE International Conference on Multimedia and Expo, 2000 (ICME), 2000, pp. 1191–1194.

[18] F. Meyer and S. Beucher, *Morphological Segmentation*, Journal of Visual Communication and Image Representation **1** (1990), no. 1, 21–46.

[19] A. Mitiche and P. Bouthemy, *Computation and Analysis of Image Motion: A Synopsis of Current Problems and Methods*, International Journal of Computer Vision **19** (1996), no. 1, 29–56.

[20] R. Mosges and S. Lavallee, *Computer Integrated Surgery - Technology and Clinical Applications* (1996), 5–19.

[21] I. Patras, E. A. Hendriks, and R. L. Lagendijk, *Video Segmentation by MAP Labeling of Watershed Segments*, IEEE Transactions on Pattern Analysis and Machine Learning **23** (2001), no. 3, 326–332.

[22] P. Salembier and M. Pardàs, *Hierarchical Morphological Segmentation for Image Sequence Coding*, IEEE Transactions on Image Processing **3** (1994), no. 5, 639–651.

[23] P. Salembier and J. Ruiz, *Connected operators based on reconstruction process for size and motion simplification*, IEEE International Conference on Acoustics, Speech, and Signal Processing, 2002, 2002, pp. 3289–3292.

[24] P. Soille and L. Vincent, *Determining Watersheds in Digital Pictures via Flooding Simulations*, Visual Communications and Image Processing, 1990, pp. 240–250. volume 1360.

[25] T. Yoshida and T. Shimosato, *Motion image segmentation using 3-D watershed algorithm*, IEEE International Conference on Image Processing, 2001, 2001, pp. 773–776.

Two approaches for orientation field segmentation based on directional morphology

Luis A. Morales-Hernández[1], Federico Manríquez-Guerrero[2] and Iván R. Terol-Villalobos[2]

[1] *Doctorado en Ingeniería, Universidad Autónoma de Querétaro, Mexico*
`luis_morah@yahoo.com`

[2] *Centro de Investigación y Desarrollo Tecnológico en Electroquímica S. C. (CIDETEQ), Parque Tecnológico Querétaro, Mexico*
`{fmanriquez,iterol}@cideteq.mx`

Abstract In the present paper, morphological approaches for segmenting orientation fields are proposed. First, it is investigated the use of directional granulometries and a quadtree structure to extract the main directional structures of the image. Then, it is proposed a method based on the concept of the line-segment and orientation functions. The line-segment function is computed from the supremum of directional erosions. This function contains the sizes of the longest lines that can be included in the structure. On the other hand, the orientation function contains their angles. Combining both functions permits the construction of a weighted partition using the watershed transformation. Finally, the elements of the partition are merged using a region adjacency graph (RAG) structure.

Keywords: directional morphology, directional granulometry, orientation fields, line-segment function, orientation function, watershed transform.

1. Introduction

Even if anisotropic structures are frequently found in many classes of images (materials, biometry images, biology, ...), few works dealing with directional analysis in morphological image processing have been carried out. From an algorithmic point of view one has [14, 15] among others, while in application some references are [5, 16]. It is in the domain of fingerprint recognition, which is today the most widely used biometric features for personal identification, where the study of directional structures based on orientation-field detection is an active subject of research [2, 7, 9]. In fact, fingerprints can be considered as a structure composed by a set of line segments (see Figure 1(a)). However, the orientation-field detection also plays a fundamental role in other domains [1, 6]. For example, in materials, the pearlite phase displays a morphology in the form of parallel lines (see Figure 1(b)) and when forming another grain, these can change of direction. In this case, field extraction from an image is a useful technique for the characterization of the pearlitic phase.

Given the interest in orientation pattern models for characterizing structures, this paper investigates the use of the mathematical morphology for modelling orientation fields. As for the human vision, computer image processing of oriented image structures often requires a bank of directional filters or template masks, each of them sensitive to a specific range of orientations. Then, one investigates first the use of directional granulometries, computed by morphological openings, using directional structuring elements. This approach allows one to determine the main directions of the structures by identifying the minima of the granulometric distribution function. In order to define a local approach, a quadtree structure is used to decompose the image with different resolution according to the levels of the tree.

After illustrating some drawbacks of using a bank of morphological filters (openings) and a quadtree structure to characterize orientation fields, one introduces an approach based on directional erosions. This method considers a local approach using the concept of line-segment function combined with the watershed transformation. In our case, the line-segment function is computed from the supremum of directional erosions. This function contains the information of the longest line segments that can be placed inside the structure. In order to know their orientation, a second image is defined by observing the construction of the line-segment function and its evolution. This second image is computed by detecting the orientation of the supremum of directional erosions. These local descriptors, for the element size and the orientation, enable the identification of the orientation fields based on the watershed transformation.

This paper is organized as follows. In Section 2, the concepts of morphological filter and directional morphology are presented. In Sections 3 and 4, a study for segmenting orientation fields based on the directional granulometry is carried out. Next, in Section 5 the notions of line-segment and orientation functions, derived from the supremum of directional erosions, are introduced. Finally, in Section 6 an approach of working with directional morphology, the watershed transform and a region adjacency graph (RAG) for segmenting orientation fields is proposed.

2. Some basic concepts of morphological filtering

In mathematical morphology one calls morphological filter all increasing and idempotent transformation [4, 12]. The basic morphological filters are the morphological opening $\gamma_{\mu B}$ and the morphological closing $\varphi_{\mu B}$ given a structuring element B (for example a square of 3×3 pixels) and an homothetic parameter μ. Let $\check{B}$ is the transposed set of B ($\check{B} = \{-x :$

$x \in B\}$), the morphological opening and closing are given by: $\gamma_{\mu B}(f) = \delta_{\mu \check{B}}(\varepsilon_{\mu B}(f))$ and $\varphi_{\mu B}(f) = \varepsilon_{\mu \check{B}}(\delta_{\mu B}(f))$, where the morphological erosion $\varepsilon_{\mu B}$ and dilation $\delta_{\mu B}$ are expressed by $\varepsilon_{\mu B}(f)(x) = \wedge\{f(y) : y \in \mu \check{B_x}\}$ and $\delta_{\mu B}(f)(x) = \vee\{f(y) : y \in \mu \check{B_x}\}$, and where $\wedge$ is the infimum and $\vee$ is the supremum.

Morphological directional transformations are characterized by two parameters. Their structuring elements are line segments L having a length (size μ) and a slope (angle α). For $\alpha \in [0, 90]$, the line segment $L(\alpha, \mu)$ is formed of the set of points $\{(x_i, y_i)\}$ computed using the following expressions:

if $0 \leq \alpha \leq 45$ then, $y_i = x_i tan(\alpha)$ for $x_i = 0, 1, \cdots, (\mu/2)cos(\alpha)$,

if $90 \geq \alpha > 45$ then, $x_i = y_i cot(\alpha)$ for $y_i = 0, 1, \cdots, (\mu/2)cos(\alpha)$,

and of the set of points $\{(-x_i, -y_i)\}$.

In this way, the structuring element is a symmetric, i.e., $L(\alpha, \mu)=\hat{L}(\alpha, \mu)$. Similar expressions can be used for $\alpha \in (90, 180]$.

For the sake of simplicity, from now on, we will denote the morphological opening $\gamma_{L(\alpha,\mu)}$ and closing $\varphi_{L(\alpha,\mu)}$, respectively, $\gamma_{\alpha,\mu}$ and $\varphi_{\alpha,\mu}$.

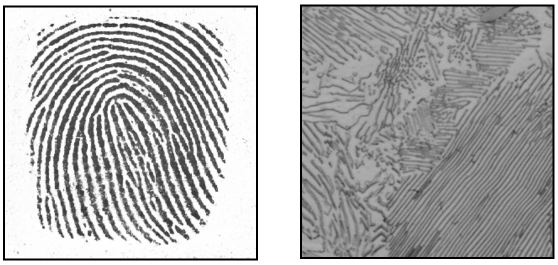

(a) (b)

Figure 1. (a) Fingerprint image. (b) Pearlitic phase image.

3. Directional granulometry

Granulometry was formalized by Matheron for binary images and extended to complete lattices by Serra [12]. Granulometry is defined as follows:

Definition 1 (Granulometry)**.** A family of openings $\{\gamma_{\mu_i}\}$ (or respectively of closings $\{\varphi_{\mu_i}\}$), where $i \in \{1, 2, \ldots n\}$, is a granulometry (respectively antigranulometry) if for all $i, j \in \{1, 2, \ldots n\}$ and for all function f,

$$\mu_i \leq \mu_j \Rightarrow \gamma_{\mu_i}(f) \geq \gamma_{\mu_j}(f) \qquad \text{(resp. } \varphi_{\mu_i}(f) \leq \varphi_{\mu_j}(f)\text{)}.$$

□

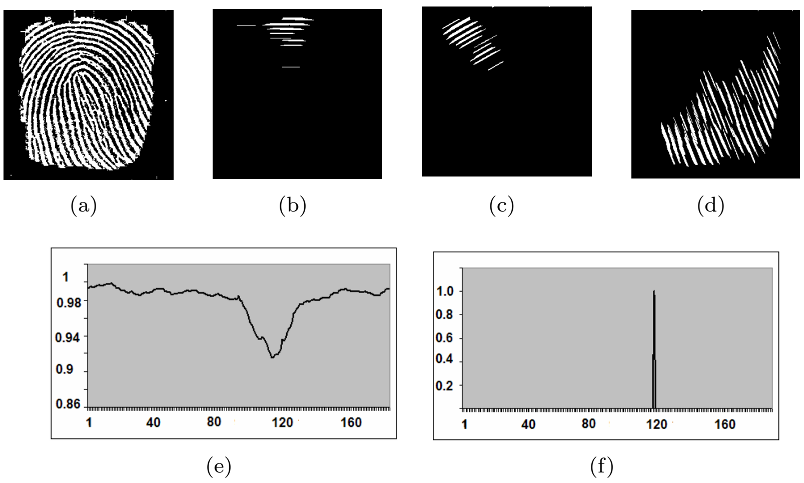


Figure 2. (a) Binary image. (b–d) Directional opening of size 80 and angles 0, 55, 112, respectively. (e) Granulometric curve computed from image (a). (f) Minima detection.

The ordering relationship implies that greater the parameter, more severe the opening (closing). The granulometric analysis of a binary or gray-level image, consists in associating with each μ_i value a measure of the image $\gamma_{\mu_i}(f)$. Two functions are associated to these transformations: the granulometric density function g and its distribution function G given respectively by:

$$g(\alpha, \lambda)(f) = (Mes(\gamma_{\alpha,\lambda+1})(f) - Mes(\gamma_{\alpha,\lambda})(f))/Mes(f),$$

$$G(\alpha, \lambda)(f) = (Mes(f) - Mes(\gamma_{\alpha,\lambda})(f))/Mes(f),$$

where Mes represents the volume for gray-level images and the area for binary images. To illustrate the use of the granulometry for detecting anisotropies inside a structure, the binary image of Figure 2(a) was computed from the gray-level image of Figure 1(a). Figures 2(b), 2(c) and 2(d), illustrate the output images computed from the image of Figure 2(a), using a directional opening of size $\mu = 80$ and angles 0, 55 and 112 degrees, respectively. Observe that this microstructure contains a main direction at approximately 112 degrees. To detect automatically the main direction in a structure one computes a granulometry as described below.

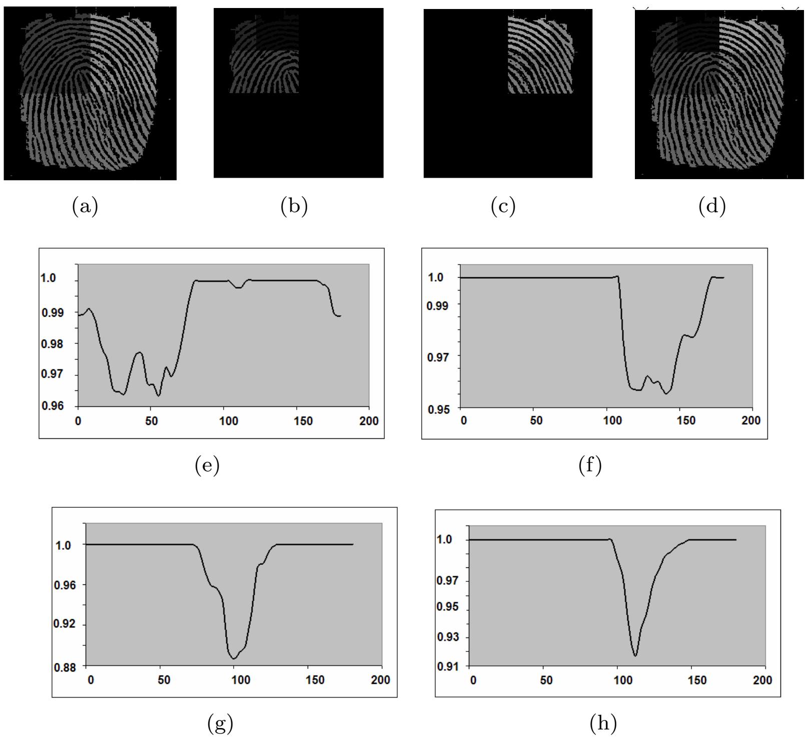


Figure 3. (a) First hierarchy of the quatree. (b, c) Second hierarchy of the quadtree. (d) Final segmentation. (e–h) Granulometric curves of the first hierarchy.

4. Directional granulometry and quadtree structure

By computing the density function $g(\alpha, \lambda)$ one obtains the portion of the structure, for a given direction α, of size λ, whereas the distribution function $G(\alpha, \lambda)$ gives the fraction of the structures greater than or equal to the length λ in the direction α. This latter function is more interesting since it permits the selection of the main structures in a given direction. Thus, instead of fixing the parameter α, the parameter λ was fixed. Figure 2(e), illustrates the distribution function of the image of Figure 2(a), for $\lambda = 80$ and $0 < \alpha < 180$. This expression permits one to know the percentage of the structure removed by the opening.

For some angles the directional opening removes all of the structure, and $G(\alpha, \lambda)(f) \approx 1$, whereas in the direction of the longest structures $G(\alpha, \lambda)(f) < 1$. The global minimum of this function permits us to determine the direction of the main structures. The minimum in Figure 2(f), was computed from the function of Figure 2(e), using morphological transformations in one-dimensional case. To carry out the minimum detection, the distribution function was scaled into the interval $[0, 255]$ in integer numbers and then, the traditional morphological tools for detecting minima in mathematical morphology were applied. In fact, the minima of the function will enable us to have a criterion to go from a global approach to a local one by means of the quadtree structure.

In the quadtree approach, the coding by regions is made by an homogeneity criterion (or criteria) that enables us to discriminate whether a square region can be considered a connected component. One starts with a square of 2^n pixels that is subdivided in four square zones. Each square zone is analyzed as a part of the original image using one or several homogeneity criteria (for example, variance, max-min values). If the homogeneity criterion (or criteria) is verified, a function value is given at all points of the square region (for example, the average of the intensity values in the square). For any square region that does not satisfy the homogeneity criterion, a similar procedure is performed in a recursive way by further dividing the square region by four.

For orientation fields, it is clear that an homogeneity criterion is given by a directional one and in our case, the minima of the distribution function are used as the criterion. If the distribution function in a square region presents only a principal minimum, then the region is considered homogeneous. In this case, the pixel values of the region are replaced by the angle of the minimum where the minimum was found. Otherwise, if the distribution function of a square region has several representative minima, then the region is devided again by four.

Figure 3(a) illustrates the approach to determine the orientation field in the image. After dividing the image by four (see Figure 3(a), the four distribution functions were computed as illustrated in Figures 3(e), 3(f), 3(g) and 3(h). In particular observe that the distribution functions of Figures 3(g) and 3(h), corresponding to the bottom right and left squares each contains only one principal minimum, while the other two squares, Figure 3(e) and Figure 3(f), contain several representative minima. Thus, these two squares were subdivided by four and their distribution functions were computed to know their directional homogeneity. Figure 3(c) illustrates the top square regions divided by four squares regions. Finally, Figure 3(d) illustrates the final hierarchy.

5. Size and orientation codification based on directional erosions

The approach described above for segmenting orientation fields has some main drawbacks. The first one is due to the fact that some regions must be processed several times (according to the hierarchy of the quadtree). A second problem is that the final segmentation will be composed by square

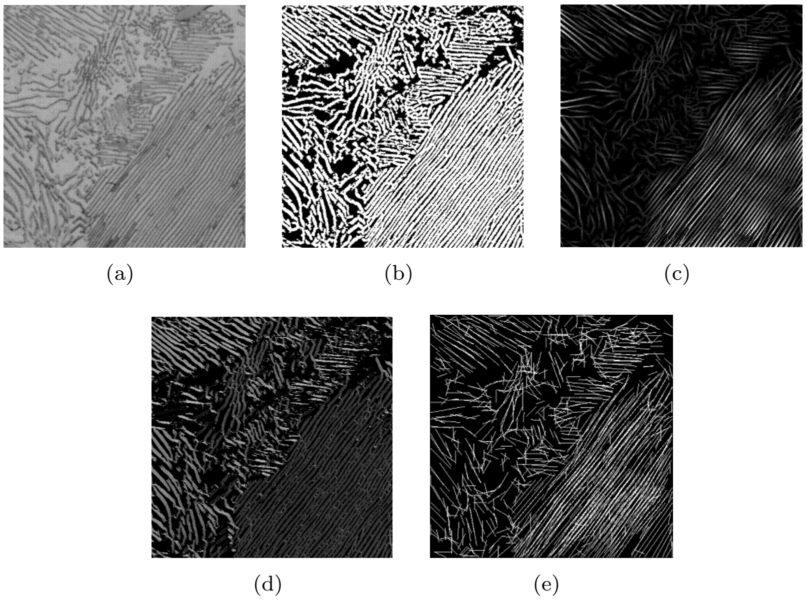


Figure 4. (a) Original image. (b) Binary image. (c, d) Line-segment and orientation functions. (e) Straight lines at the regional maxima of the line-segment function.

regions (or the union of square regions) which is not a real representation of image structures.

In this section and in the following ones we will look for another approach where the connectivity notion plays a fundamental role for segmenting the orientation fields. In fact, it is well-known that the notion of connectivity is linked to the intuitive idea of segmentation task, where the objective is to split the connected components in a set of elementary shapes that will be processed separately. Then, the problem lies in determining what a connected component is for an image such as those illustrated in Figure 1(a) and in Figure 1(b). Given that, we will look for another approach where the information of scales and directions of the structures of the image are easily accessible. Two functions that codify the size and the orientation are introduced below.

The idea for codifying size structure come from the notion of the distance function $D_X(x)$ that is a transformation that associates with each pixel x of a set X its distance from the background. The distance function can be computed by successive erosions of the set X. Let us now build a new function derived from the notion of distance function. The goal of building this function consists in codifying the size information in such a way that local directional information can be accessed from each point of the function.

This codification of the size information will be used to build a local approach for detecting orientation fields on an image. This function, which we call line-segment function Dm, is computed by using the supremum of directional erosions. To stock the size information for all λ values, a gray-level image Dm is used. Thus, one begins with a small structuring element by taking into account all orientations to compute the set $\sup_{\alpha\in[0,180]}\{\varepsilon_{L(\lambda,\alpha)}(X)\}$. Then one increases $Dm(x)$ by one at all points x belonging to the set $\sup_{\alpha\in[0,180]}\{\varepsilon_{L(\lambda,\alpha)}(X)\}$, and one continues the procedure by increasing the size of the structuring element until the structure is completely removed. This means that the procedure continues until one has a $\lambda_{\max}$ value such that $\sup_{\alpha\in[0,180]}\{\varepsilon_{L(\lambda_{\max},\alpha)}(X)\} = \emptyset$. The maxima of the function Dm are the loci of longest structuring elements. Thus, one knows the position of the largest structuring elements that can be included completely in the structure.

However, the angles of these structuring elements (line segments) are not accessible from the image Dm. Therefore, one stocks the directions of the line segments in a second image Om, called orientation function, when the line-segment function is computed.

A real example (pearlitic phase) is shown in Figure 4(b), which is the binary image of that in Figure 4(a). The image of Figure 4(c) illustrates the line-segment function Dm whereas the image of Figure 4(d) shows the orientation Om function, computed from the binary image of Figure 4(b). Now, these functions can be now used for computing the line segments that characterize the structure. To illustrate the information contained in these images, the maxima of Dm were computed for obtaining the loci of the maximal structuring elements. Next, a line segment was placed at each maximum point x, with an angle given by $Om(x)$. The longest line segments in the image are illustrated in Figure 4(e). The line-segment function and its associated orientation image containing the angles, serve to suggest a method for segmenting images of orientation fields.

6. Image segmentation using directional morphology and the watershed transformation

Image segmentation is one of the most interesting problems in image processing and analysis. The main goal in image segmentation consists in extracting the regions of greatest interest in the image [3, 8]. A segmentation method must allow the introduction of specific criteria to obtain the desired regions (e.g., gray level, contrast, size, shape, texture, etc). In mathematical morphology, the watershed-plus-marker approach is the traditional image segmentation method [8]. This method has proved to be an efficient tool in many image-segmentation problems. Here, the watershed will be applied directly for obtaining a fine partition, and then a systematic merging process will be applied to obtain the final segmentation.

Figure 5(a) shows the inverse image of image Dm in Figure 4(c), while

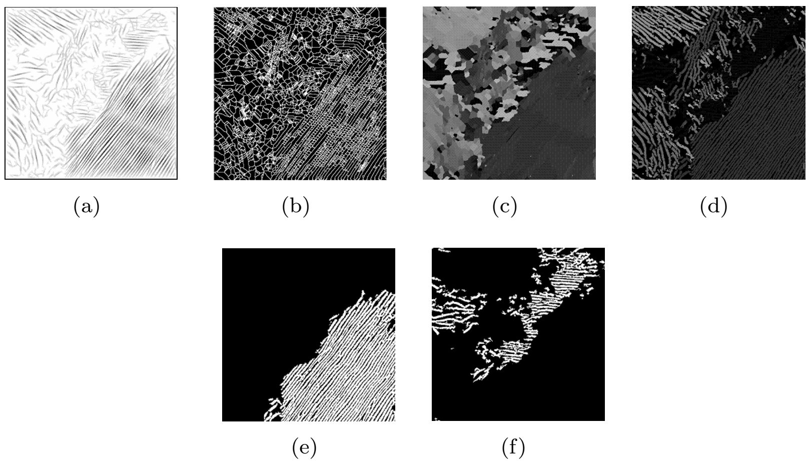


Figure 5. (a) Inverse line-segment function. (b) Watershed image. (c) Weigthed catchment basins. (d) Segmented image. (e, f) Connected components.

Figure 5(b) illustrates its watershed image. To realize the merging process it is preferable to work with the catchment basins associated with the watershed image. Figure 5(c) shows the catchment basins weighted by the values of the angles of the regional maxima of the image Om of Figure 4(d). Now, by analyzing a region of the image of Figure 5(c), one can identify the neighboring regions with more-or-less similar orientations. In order to take into account their neighborhood relationships, a region adjacency graph (RAG) must be computed. In fact, the RAG simplifies the merging process. We have chosen the method proposed in [13] for the merging process. Let us introduce some concepts concerning graphs.

A graph is a pair made of a set V of vertices and a family of arcs.

Here, one considers the case of a graph without loops. This means that there are no arcs connecting a vertex to itself. In the general case, arcs are oriented; in this work, however, one takes the case of non-oriented graphs: if there exists an arc joining vertex v to vertex v', then there also exists an arc joining v' to v.

A vertex v' is said to be a neighbor of a given vertex v if there exists an arc joining v to v'.

Thus, one way to represent a RAG consists of associating a vertex to each region and an edge to each pair of adjacent regions. By definition the RAG provides a *simple connectivity view* of the image. Beyond this simple connectivity view this graph also gives a *high-level connectivity view* of the image. Consider three regions A, B and C of an image. Thus, if two regions A and B are adjacent, and also the regions B and C are adjacent, but A and C are not adjacent, that leads us to consider that regions A and C have a second order connectivity relationships.

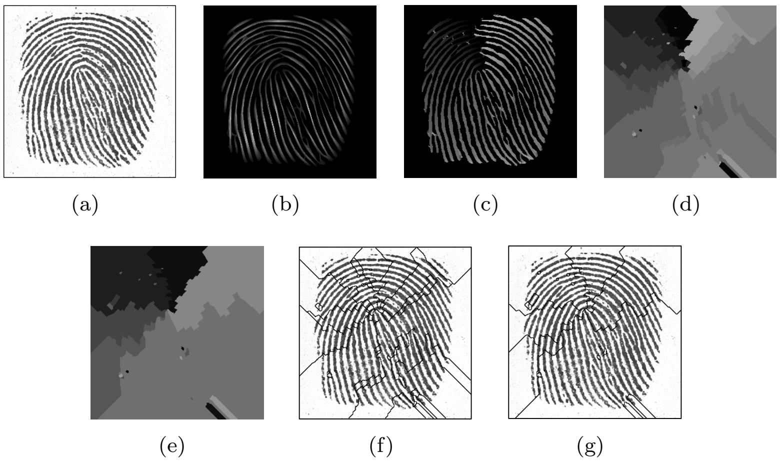


Figure 6. (a) Original image. (b, c) Line-segment and orientation functions. (d, e) Images computed after the merging process using criteria values of 10 and 20 degrees, repectively. (f, g) Contours imposed to the original image.

In fact, the simple connectivity view contains inherently all *high-level connectivity relationships* of the image. Each vertex v_i corresponds a region R_i with orientation values (for example, $\overrightarrow{\mu}_i$ and $\overrightarrow{\sigma}_i$ mean value and variance value of the region) representative of the orientation distribution of this region. Each edge e_{ij} represents a pair of adjacent regions $\{R_i, R_j\}$ with a corresponding orientation distance $d(R_i, R_j)$, which can be used to compare the orientation distribution of these two regions. In our case, the computation of the RAG, using the angles of the regions, guides the subsequent merging of regions and provides a complete description of the neighborhoods. The RAG graph is constructed by use of the catchment basins of the image of Figure 5(c).

One takes a point from each minimum of the inverse line-segment function for representing each catchment basin. Remember that the inverse distance function is used. Since the graph under study is a valued, one must introduce some numerical values. Each edge is then assigned a value given by the absolute value of the difference between the angles of two neighboring regions, computed from the orientations image. The neighborhood graph of the maxima of the line-segment function and the directional function synthesize the directional field of the image. Two vertices of the graph are linked by an edge if the catchment basins are neighbors, and the value of the edge represents the directional similarity. One the regions are codified on a graph, we can compute the orientation fields based on the valued graph.

The following method (see [13]) for reducing the numbers of regions was carried out.

1. Each border has assigned an angle distance between the two regions it separates.
2. The borders are sorted in increasing order.
3. Two regions separated by the least value border are merged.
4. Step 2 is repeated until the criterion cannot be satisfied.

We illustrate the method by identifying the adjacent regions with more-or-less similar orientation by considering the image of Figure 4(a), a micrograph of the pearlite structure in steel. To achieve such a goal, one merges the vertices (catchment basins) with a difference of angles smaller than or equal to a given angle value $d(R_i, R_j) = |angle(R_i) - angle(R_j)| \leq \theta$. Figure 5(d), shows the segmented image of the orientation function of Figure 5(c), while Figures 5(e) and 5(f) show some connected components after the merging process using angles difference criterion θ of 20. The same approach was carried out with the fingerprint image shown in Figure 6(a). Figures 6(b) and 6(c) illustrate the line-segment function and the orientation function, whereas in Figures 6(d) and 6(e), one shows the images computed after the merging process using criteria values θ of 10 and 20, respectively. Finally, Figures 6(f) and 6(g), illustrate the contours imposed to the original image.

7. Conclusion and future works

This paper has shown the possibilities for application of morphological directional transformations to segment images with orientation fields. Initially, one investigates the directional granulometries and the notion of quadtree structure. The quadtree is used to describe a class of hierarchical data structures; thus it permits one to classify the orientation fields at different scales. After some drawbacks of this approach are illustrated, one considers a second local approach. This approach involves a local analysis using the notions of the line-segment and orientation functions proposed in this paper. The maxima of the line-segment function were used for computing the loci of maximal structuring elements, and the orientation function was used to obtain the angles of the line segments. These pairs of local parameters enable us to produce a good description of the image by means of line segments. Then, a partition of the image may be computed by means of the catchment basins associated with the watershed transform. This enables us to realize a neighborhood analysis, using a RAG structure, in order to merge adjacent regions of the partition according to appropriate criteria, thus segmenting the images into orientation fields. The results based on the algorithms presented in this paper show the good performance of the approach. Future work will be in the direction of seeking for an optimal segmentation based on lattice approach for morphological image segmentation proposed recently by Serra [11].

Acknowledgements

We wish to thank the anonymous reviewer whose comments will help us for future works. The author Luis Morales thanks the government agency CONACyT for the financial support. The author I. Terol would like to thank Diego Rodrigo and Darío T. G. for their great encouragement. This work was funded by the government agency CONACyT, Mexico.

References

[1] C. Bahlmann, *Directional features in online handwriting recognition*, Pattern Recognition **39** (2006), 115–125.

[2] R. Cappelli and A. Lumini, *Fingerprint classification by directional image partitioning*, IEEE Trans. on Pattern Anal. Machine Intell **21** (1999), no. 5, 402–421.

[3] J. Crespo, R. Schafer, J. Serra, C. Meyer, and C. Gratin, *A flat zone approach: A general low-level region merging segmentation method*, Signal Proces. **62** (1997), 37–60.

[4] H. J. A. M. Heijmans, *Morphological image operators*, Academic, Boston, 1994.

[5] D. Jeulin and M. Kurdy, *Directional mathematical morphology for oriented image restoration and segmentation*, Acta Stereologica **11** (2001), 545–550.

[6] J. K. Lee, T. S. Newman, and Gary G. A., *Oriented connectivity-based method for segmenting solar loops*, Pattern Recognition **39** (2006), 246–259.

[7] J. Li, W. Y. Yau, and Wang H., *Constrained nonlinear models of fingerprint orientations with prediction*, Pattern Recognition **39** (2006), 102–114.

[8] J. Meyer and S. Beucher, *Morphological segmentation*, J. Vis. Comm. Image Represent. **1** (1990), 21–46.

[9] C. H. Park, J. J. Lee, M. J. T. Smith, and K. H. Park, *Singular point detection by shape analysis of directional fields in fingerprints*, Pattern Recognition **39** (2006), 839–855.

[10] J. Serra, *Image analysis and mathematical morphology*, Academic, London, 1982.

[11] ———, *A lattice approach to image segmentation*, J. Mathematical Imaging and Vision **24** (2006), no. 1, 83–130.

[12] ———, *Image analysis and mathematical morphology*, Academic, London, 1988.

[13] L. Shafarenko, M. Petrou, and Kittler J., *Automatic watershed segmentation of randomly textured color images*, IEEE Trans. on Image Processing **6** (1997), no. 11, 1530–1544.

[14] P. Soille, E. J. Breen, and R. Jones, *Recursive implementation of erosions and dilations along discrete lines at arbitrary angles*, IEEE Trans. on Pattern Anal. Machine Intell **18** (1996), no. 5, 562–567.

[15] P. Soille and H. Talbot, *Directional morphological filtering*, Pattern Recognition **23** (2001), no. 11, 1313–1329.

[16] A. Tuzikov, P. Soille, D. Jeulin, and P. Vermeulen, *Extraction of grid patterns on stamped metal sheets using mathematical morphology*, International Conference on Pattern Recognition, 1 (1992), pp. 425–428.

Design of robust pattern classifiers based on optimum-path forests

João P. Papa[1], Alexandre X. Falcão[1], Paulo A. V. Miranda[‡, 1], Celso T. N. Suzuki[§, 1] and Nelson D. A. Mascarenhas[2]

[1] *Instituto de Computação (IC), Universidade Estadual de Campinas (Unicamp), SP, Brazil* {jpaulo,afalcao}@ic.unicamp.br

[2] *Departamento de Computação, Universidade Federal de São Carlos (UFSCar), SP, Brazil*
nelson@dc.ufscar.br

Abstract We present a supervised pattern classifier based on *optimum path forest.* The samples in a training set are nodes of a complete graph, whose arcs are weighted by the distances between sample feature vectors. The training builds a classifier from key samples (prototypes) of all classes, where each prototype defines an optimum path tree whose nodes are its strongest connected samples. The optimum paths are also considered to label unseen test samples with the classes of their strongest connected prototypes. We show how to find prototypes with none classification errors in the training set and propose a learning algorithm to improve accuracy over an evaluation set. The method is robust to outliers, handles non-separable classes, and can outperform support vector machines.

Keywords: supervised classifiers, image foresting transform, image analysis, morphological pattern recognition.

1. Introduction

Pattern classification methods are generally divided into supervised and unsupervised according to their learning algorithms [9]. Unsupervised techniques assume no knowledge about the classes (labels) of the samples in the training set, while these labels are exploited in supervised techniques.

We propose a method to project supervised pattern classifiers based on *optimum path forests* (OPF). The design of an OPF classifier is based on labeled samples from training and evaluation sets. A test set with unseen samples is used to assess the performance of the classifier.

The training samples are nodes of a complete graph in the sample feature space (all pairs of nodes are connected by one arc). See Figure 1(a). The arcs

[‡]pavmbr@yahoo.com.br
[§]celso.suzuki@gmail.com

are weighted by the distance between the feature vectors of their nodes. A set of prototypes (key samples) is obtained from the training set. We define a *path-cost function* based on arc weights, which assigns to any path in the graph the cost of considering all samples along the path as belonging to a same class (e.g., function f_{max} which assigns the maximum arc weight along the path). We then apply the IFT algorithm [10] to partition the graph into an optimum path forest rooted at the prototypes (Figure 1(b)). That is, the prototypes compete among themselves and each prototype defines an optimum path tree, whose nodes are samples more strongly connected to that prototype than to any other root, according to that path-cost function. The training essentially consists of building this optimum path forest, where the samples in a given optimum path tree are assumed to have the same label of their root prototype.

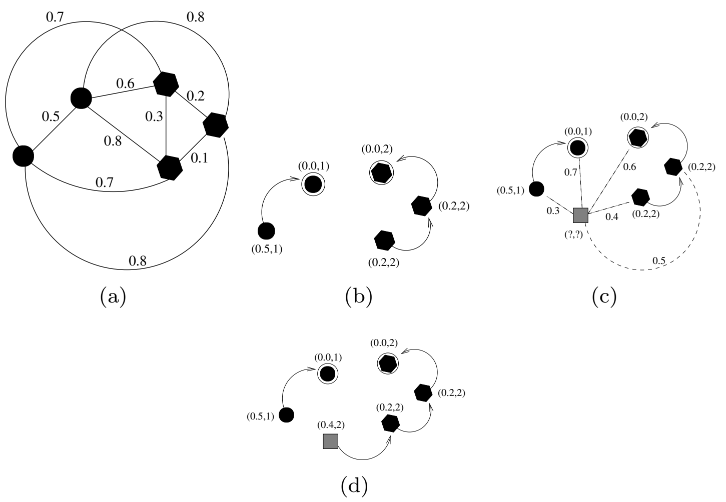


Figure 1. (a) Complete weighted graph for a simple training set. (b) Resulting optimum-path forest from (a) for f_{max} and two given prototypes (circled nodes). The entries (x, y) over the nodes are, respectively, cost and label of the samples. (c) Test sample (gray square) and its connections (dashed lines) with the training nodes. (d) The optimum path from the most strongly connected prototype, its label 2, and classification cost 0.4 are assigned to the test sample.

The classification of a test sample evaluates the optimum paths from the prototypes to this sample incrementally, as though it were part of the graph (Figure 1(c)). The optimum path from the most strongly connected prototype, its label and path cost (classification cost) are assigned to the test sample (Figure 1(d)). Note the difference between an OPF classifier with $f_{\max}$ and the nearest neighbor approach [9]. The test sample is assigned to a given class, even when its closest labeled sample is from another class. The same rule is used to classify evaluation samples.

Before testing, we propose a learning algorithm which replaces new samples of the evaluation set by *irrelevant* samples of the training set. Very often real problems limit the training set size. The learning algorithm aims to improve accuracy with this limitation. When an evaluation sample is classified, it is assigned to some optimum path in the graph. The training samples of this path have their numbers of right or wrong classifications added by one, depending on the classification result. The irrelevant samples are those with the number of wrong classifications higher than the number of right classifications. At each iteration, the learning algorithm creates new evaluation and training sets and recomputes prototypes and optimum-path forests. These prototypes guarantee none classification errors in the training set and usually improve the accuracy over the evaluation sets. The presence of outliers (samples of a given class that fall inside the region of another class) usually degrades the project of any classifier. Outliers usually become irrelevant prototypes and are moved out from the training set. The number of prototypes will not necessarily increase during learning and the most representative are usually in the frontiers between classes. The method handles non-separable classes by estimating key prototypes within the intersection regions.

Section 2 discusses related works. The OPF classifier is presented in Section 3 and Section 4 presents its learning algorithm, which outputs the last designed classifier and a learning curve showing the accuracy values of the designed classifiers along its iterations. In Section 5, we compare the OPF classifier with support vector machines (SVM) [3]. This comparison uses databases with outliers and non-separable multiple classes, being two databases from image analysis. One contains voxels from white and gray matters in magnetic resonance images of the human brain and the other contains 2D shapes from binary images. Conclusions and future works are discussed in Section 6.

2. Related works

Graph-based approaches for pattern classification are usually unsupervised. Zahn [19] proposed an approach that computes a minimum spanning tree (MST) in the graph and removes inconsistent arcs to form clusters. Arc removal in the MST can also produce hierarchical solutions for clustering, such as the popular single-linkage approach [11]. Other clustering techniques have been formulated as a graph-cut problem [17] with application to image segmentation, where the graph does not need to be complete. More recently, graph-cut techniques have also been used for learning [2]. Essentially, graph-based clustering methods aim to partition the graph into components (clusters), such that each component contains only samples of a same class. However, there is no guarantee that the samples in a given cluster belong to the same class, and it is hard to assign these samples to their correct class without any prior knowledge.

Supervised approaches usually exploit prior knowledge to teach the machine how to solve the problem. Artificial neural networks (ANN) [12] and support vector machines (SVM) [3] are among the most actively pursued approaches in the last years. An ANN multi-layer perceptron (ANN-MLP), trained by backpropagation for example, is an unstable classifier. Its accuracy may be improved at the computational cost of using multiple classifiers and algorithms (e.g., bagging and boosting) for training classifier collections [12]. However, it seems that there is an unknown limit in the number of classifiers to avoid an undesirable degradation in accuracy [16]. ANN-MLP assumes that the classes can be separated by hyperplanes in the feature space. Such assumption is unfortunately not valid in practice. SVM was proposed to overcome the problem by assuming it is possible to separate the classes in a higher dimensional space by optimum hyperplanes. Although SVM usually provides reasonable accuracies, its computational cost rapidly increases with the training set size and the number of support vectors. [18] proposed a method to reduce the number of support vectors in the multiple-classes problem. Their approach suffers from slow convergence and higher computational complexity, because they first minimize the number of support vectors in several binary SVMs, and then share these vectors among the machines. [15] presented a method to reduce the training set size before computing the SVM algorithm. Their approach aims to identify and remove samples likely related to non-support vectors. However, in all SVM approaches, the assumption of separability may also not be valid in any space of finite dimension [6].

The role of the prototypes for the OPF classifier is very similar to the importance of the support vectors for SVM. Considering this together with the fact that SVM is among the best approaches for supervised pattern classification, we have chosen the SVM code in [4] with a Gaussian kernel and parameters obtained by cross validation for comparison.

3. Optimum path classifier

Let Z_1, Z_2, and Z_3 be training, evaluation, and test sets with $|Z_1|$, $|Z_2|$, and $|Z_3|$ samples such as points or image elements (e.g., pixels, voxels, shapes). Let $\lambda(s)$ be the function that assigns the correct label i, $i = 1, 2, \ldots, c$, from class i to any sample $s \in Z_1 \cup Z_2 \cup Z_3$. Z_1 and Z_2 are labeled sets used to the design of the classifier. The applications usually impose an upper limit in $|Z_1|$, then the role of Z_2 is to improve the accuracy of the classifier by interchanging samples with Z_1. Z_3 is used to assess the performance of the classifier and it is kept unseen during the project.

Let $S \subset Z_1$ be a set of prototypes of all classes (i.e., key samples that best represent the classes). Let v be an algorithm which extracts n attributes (color, shape or texture properties) from any sample $s \in Z_1 \cup Z_2 \cup Z_3$ and returns a vector $\vec{v}(s) \in Re^n$. The distance $d(s, t)$ between two samples, s and t, is the one between their feature vectors $\vec{v}(s)$ and $\vec{v}(t)$. One can use any valid metric (e.g., Euclidean) or a more elaborated distance algorithm [1].

Our problem consists of using S, (v, d), Z_1 and Z_2 to project an optimal classifier which can predict the correct label $\lambda(s)$ of any sample $s \in Z_3$. We propose a classifier which creates a discrete optimal partition of the feature space such that any sample $s \in Z_3$ can be classified according to this partition. This partition is an optimum path forest (OPF) computed in $\Re^n$ by the image foresting transform (IFT) algorithm [10].

Let (Z_1, A) be a complete graph whose the nodes are the samples in Z_1 and any pair of samples defines an arc in $A = Z_1 \times Z_1$ (Figure 1(a)). The arcs do not need to be stored and so the graph does not need to be explicitly represented. A path is a sequence of distinct samples $\pi = \langle s_1, s_2, \ldots, s_k\rangle$, where $(s_i, s_{i+1}) \in A$ for $1 \leq i \leq k-1$. A path is said *trivial* if $\pi = \langle s_1\rangle$. We assign to each path π a cost $f(\pi)$ given by a path-cost function f. A path π is said optimum if $f(\pi) \leq f(\pi')$ for any other path π', where π and π' end at a same sample s_k. We also denote by $\pi \cdot \langle s, t\rangle$ the concatenation of a path π with terminus at s and an arc (s, t).

The OPF algorithm may be used with any *smooth* path-cost function which can group samples with similar properties [10]. A function f is smooth in (Z_1, A) when for any sample $t \in Z_1$, there exists an optimum path π ending at t which either is trivial, or has the form $\tau \cdot \langle s, t\rangle$ where

- $f(\tau) \leq f(\pi)$,
- τ is optimum,
- for any optimum path τ' ending at s, $f(\tau' \cdot \langle s, t\rangle) = f(\pi)$.

We will address the path-cost function f_{max}, because of its theoretical properties for estimating optimum prototypes:

$$f_{max}(\langle s\rangle) = \begin{cases} 0 & \text{if } s \in S, \\ +\infty & \text{otherwise,} \end{cases}$$
$$f_{max}(\pi \cdot \langle s, t\rangle) = \max\{f_{max}(\pi), d(s, t)\}, \tag{1}$$

such that $f_{max}(\pi)$ computes the maximum distance between adjacent samples in π, when π is not a trivial path.

The OPF algorithm assigns one optimum path $P^*(s)$ from S to every sample $s \in Z_1$, forming an optimum path forest P (a function with no cycles which assigns to each $s \in Z_1 \backslash S$ its predecessor $P(s)$ in $P^*(s)$ or a marker *nil* when $s \in S$, as shown in Figure 1(b)). Let $R(s) \in S$ be the root of $P^*(s)$ which can be reached from $P(s)$. The OPF algorithm computes for each $s \in Z_1$, the cost $C(s)$ of $P^*(s)$, the label $L(s) = \lambda(R(s))$, and the predecessor $P(s)$, as follows.

Algorithm 1. OPF.

INPUT:	*A λ-labeled training set Z_1, prototypes $S \subset Z_1$ and the pair (v, d) for feature vector and distance computations.*
OUTPUT:	*Optimum path forest P, cost map C and label map L.*
AUXILIARY:	*Priority queue Q and cost variable cst.*

```
1.  For each $s \in Z_1 \backslash S$, set $C(s) \leftarrow +\infty$.
2.  For each $s \in S$, do
3.    $C(s) \leftarrow 0$, $P(s) \leftarrow nil$, $L(s) \leftarrow \lambda(s)$, and insert $s$ in $Q$.
4.  While $Q$ is not empty, do
5.      Remove from $Q$ a sample $s$ such that $C(s)$ is minimum.
6.      For each $t \in Z_1$ such that $t \neq s$ and $C(t) > C(s)$, do
7.          Compute $cst \leftarrow \max\{C(s), d(s,t)\}$.
8.          If $cst < C(t)$, then
9.              If $C(t) \neq +\infty$, then remove $t$ from $Q$.
10.             $P(t) \leftarrow s$, $L(t) \leftarrow L(s)$, $C(t) \leftarrow cst$, and insert $t$ in $Q$.
```

Lines 1–3 initialize maps and insert prototypes in Q. The main loop computes an optimum path from S to every sample s in a non-decreasing order of cost (Lines 4–10). At each iteration, a path of minimum cost $C(s)$ is obtained in P when we remove its last node s from Q (Line 5). Ties are broken in Q using first-in-first-out policy. That is, when two optimum paths reach an ambiguous sample s with the same minimum cost, s is assigned to the first path that reached it. Note that $C(t) > C(s)$ in Line 6 is false when t has been removed from Q and, therefore, $C(t) \neq +\infty$ in Line 9 is true only when $t \in Q$. Lines 8–10 evaluate if the path that reaches an adjacent node t through s is cheaper than the current path with terminus t and update the position of t in Q, $C(t)$, $L(t)$ and $P(t)$ accordingly.

The OPF algorithm for f_{max} is an "IFT-watershed transform" [13] computed in the n-dimensional feature space. Apart from this extension, the most significant contributions are the training and learning processes which find optimum prototypes (markers) in the frontier between classes and avoid *outliers* (samples of a given class that fall inside the region of another class in the feature space) in the training set, increasing the accuracy of the classifier.

The label $L(s)$ may be different from $\lambda(s)$, leading to classification errors in Z_1. The training finds prototypes with none classification errors in Z_1.

3.1 Training

We say that S^* is an optimum set of prototypes when Algorithm 1 propagates the labels $L(s) = \lambda(s)$ for every $s \in Z_1$. Set S^* can be found by exploiting the theoretical relation between *Minimum Spanning Tree* (MST) [7] and optimum path tree for f_{max}. The training essentially consists of finding S^* and an OPF classifier rooted at S^*.

By computing an MST in the complete graph (Z_1, A), we obtain a connected acyclic graph whose nodes are all samples in Z_1 and the arcs are undirected and weighted by the distance d between the adjacent sample feature vectors (Figure 2(a)). This spanning tree is optimum in the sense that the sum of its arc weights is minimum as compared to any other spanning tree in the complete graph. In the MST, every pair of samples is connected by a single path which is optimum according to f_{max}. That is, for any given sample $s \in Z_1$, it is possible to direct the arcs of the MST such that the result will be an optimum path tree P for f_{max} rooted at s.

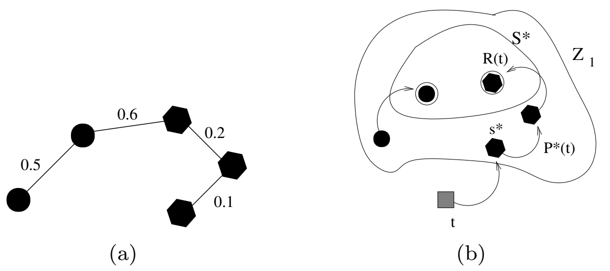


Figure 2. (a) MST of the graph shown in Figure 1a where the optimum prototypes share the arc of weight 0.6. (b) The classification of the test sample (gray square) t as in Figure 1c assigns the optimum path $P^*(t)$ from $R(t) \in S^*$ to t passing through s^*.

The optimum prototypes are the closest elements in the MST with different labels in Z_1. By removing the arcs between different classes, their adjacent samples become prototypes in S^* and Algorithm 1 can compute an optimum path forest with none classification errors in Z_1 (Figure 1(b)), which can be explained by the theoretical relation between minimum spanning trees and the optimum path tree obtained by OPF with f_{max} [8]. Note that, a given class may be represented by multiple prototypes (i.e., optimum path trees) and there must exist at least one prototype per class.

3.2 Classification

For any sample $t \in Z_3$, we consider all arcs connecting t with samples $s \in Z_1$, as though t were part of the graph (Figure 1(c)). Considering all possible paths from S^* to t, we wish to find the optimum path $P^*(t)$ from S^* and label t with the class $\lambda(R(t))$ of its most strongly connected prototype $R(t) \in S^*$ (Figure 2(b)). This path can be identified incrementally, by evaluating the optimum cost $C(t)$ as

$$C(t) = \min\{\max\{C(s), d(s,t)\}\}, \ \forall s \in Z_1. \tag{2}$$

Let the node $s^* \in Z_1$ be the one that satisfies the above equation (i.e., the predecessor $P(t)$ in the optimum path $P^*(t)$). Given that $L(s^*) = \lambda(R(t))$, the classification simply assigns $L(s^*)$ as the class of t. An error occurs when $L(s^*) \neq \lambda(t)$.

Similar procedure is applied for samples in the evaluation set Z_2. In this case, however, we would like to use samples of Z_2 to learn the distribution of the classes in the feature space and improve the performance of the OPF classifier.

4. Learning Algorithm

The performance of the OPF classifier improves when the closest samples from different classes are included in Z_1, because the method finds prototypes that will work as sentinels in the frontier between classes. We propose a learning algorithm to identify better prototypes from samples of Z_2 that have never been in Z_1 (Algorithm 2).

Algorithm 2. LEARNING.

INPUT: *Training and evaluation sets labeled by λ, Z_1 and Z_2, number T of iterations, and the pair (v, d) for feature vector and distance computations.*

OUTPUT: *Learning curve $\mathcal{L}$ and the last OPF classifier, represented by the predecessor map P, cost map C, and label map L.*

AUXILIARY: *False positive and false negative arrays, FP and FN, of sizes c, lists, LI and LE, of irrelevant samples and error samples for each class, arrays for the number of right and wrong classifications, NR and NW, of sizes $|Z_1|$, variables r for sample, and set TR to avoid samples of Z_2 return to Z_1.*

```
1.  $TR \leftarrow \emptyset$.
2.  For each iteration $I = 1, 2, \ldots, T$, do
3.      $TR \leftarrow TR \cup Z_1$.
4.      Compute $S^* \subset Z_1$ as in Section 3.1 and $P$, $L$, $C$ by Algorithm 1.
5.      For each sample $s \in Z_1$, do $NR(s) \leftarrow 0$ and $NW(s) \leftarrow 0$.
6.      For each class $i = 1, 2, \ldots, c$, do
7.          $FP(i) \leftarrow 0$, $FN(i) \leftarrow 0$, $LI(i) \leftarrow \emptyset$ and $LE(i) \leftarrow \emptyset$.
8.      For each sample $t \in Z_2$, do
9.          Find $s^* \in Z_1$ that satisfies Equation 2 and set $r \leftarrow s^*$.
10.         If $L(s^*) \neq \lambda(t)$, then
11.             $FP(L(s^*)) \leftarrow FP(L(s^*)) + 1$.
12.             $FN(\lambda(t)) \leftarrow FN(\lambda(t)) + 1$.
13.             if $t \notin TR$, then $LE(\lambda(t)) \leftarrow LE(\lambda(t)) \cup \{t\}$.
14.             While $r \neq nil$, do
15.                 $NW(r) \leftarrow NW(r) + 1$ and $r \leftarrow P(r)$.
16.         Else
17.             While $r \neq nil$, do
18.                 $NR(r) \leftarrow NR(r) + 1$ and $r \leftarrow P(r)$.
19.     Compute $\mathcal{L}(I)$ by Equation 5.
20.     For each $s \in Z_1$, do
21.         If $NW(s) > NR(s)$, then
22.             $LI(\lambda(s)) \leftarrow LI(\lambda(s)) \cup \{s\}$.
23.     For each class $i = 1, 2, \ldots, c$, do
24.         While $|LI(i)| > 0$ and $|LE(i)| > 0$, do
25.             $LI(i) \leftarrow LI(i) \backslash \{s\}$ and $LE(i) \leftarrow LE(i) \backslash \{t\}$.
26.             Replace $s \in Z_1$ by $t \in Z_2$.
27.         While $|LI(i)| > 0$, do
28.             $LI(i) \leftarrow LI(i) \backslash \{s\}$.
29.             Find $t \in Z_2 \backslash TR$, with $\lambda(t) = i$, and replace it by $s \in Z_1$.
30. Compute $S^* \subset Z_1$ as in Section 3.1 and $P$, $L$, $C$ by Algorithm 1.
```

Firstly we give preference to replace *irrelevant* samples of Z_1 by errors in Z_2, and secondly other samples of Z_2 are replaced by *irrelevant* samples of Z_1. In both cases, we never let a sample of Z_2 return to Z_1. If the application did not impose any limitation in $|Z_1|$, the prototypes could be found from $Z_1 \cup Z_2$ with none classification errors in both sets. The learning algorithm essentially tries to identify these prototypes from a few iterations of classification over Z_2.

The algorithm outputs a *learning curve* over T iterations (Lines 2–29), which reports the accuracy values of each instance of classifier during learning, and the final OPF classifier. Lines 4–7 execute training and initialize the auxiliary arrays and lists. The classification of each sample $t \in Z_2$ is performed in Lines 8–18, updating auxiliary arrays. The condition in Line 10 indicates that t is misclassified.

In order to define irrelevant samples, we consider all right and wrong classifications in Z_2. When $t \in Z_2$ is correctly/incorrectly classified, we add one to the number of right/wrong classifications, $NR(r)$ or $NW(r)$, of every sample $r \in Z_1$ in the optimum path $P^*(t)$ from $R(t) \in S^*$ to s^* (Lines 14–18). Additionally, Lines 11–13 update the number of false positive and false negative arrays, FP and FN, for accuracy computation, and insert t in the list $LE(\lambda(t))$ of errors if t has never been in Z_1 ($t \notin TR$).

Line 19 computes the accuracy at iteration I and stores it in the learning curve $\mathcal{L}$. The accuracy $\mathcal{L}(I)$ of a given iteration I, $I = 1, 2, \ldots, T$, is measured by taking into account that the classes may have different sizes in Z_2 (similar definition is applied for Z_3). Let $NZ_2(i)$, $i = 1, 2, \ldots, c$, be the number of samples in Z_2 from each class i. We define

$$e_{i,1} = \frac{FP(i)}{|Z_2| - |NZ_2(i)|} \quad \text{and} \quad e_{i,2} = \frac{FN(i)}{|NZ_2(i)|}, \; i = 1, \ldots, c \tag{3}$$

where $FP(i)$ and $FN(i)$ are the false positives and false negatives, respectively. That is, $FP(i)$ is the number of samples from other classes that were classified as being from the class i in Z_2, and $FN(i)$ is the number of samples from the class i that were incorrectly classified as being from other classes in Z_2. The errors $e_{i,1}$ and $e_{i,2}$ are used to define

$$E(i) = e_{i,1} + e_{i,2}, \tag{4}$$

where $E(i)$ is the partial sum error of class i. Finally, the accuracy $\mathcal{L}(I)$ of the classification is written as

$$\mathcal{L}(I) = \frac{2c - \sum_{i=1}^{c} E(i)}{2c} = 1 - \frac{\sum_{i=1}^{c} E(i)}{2c}. \tag{5}$$

Lines 20–22 identify as irrelevant samples in Z_1 those with number of incorrect classifications higher than the number of correct classifications.

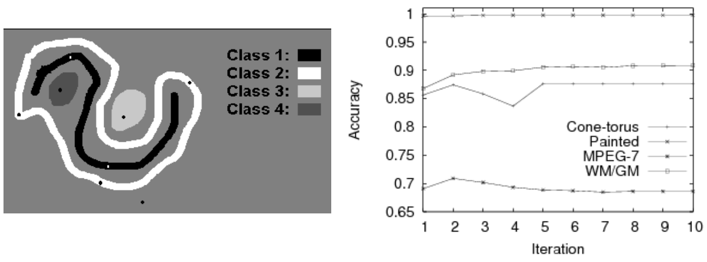
Lines 23–29 remove elements from the lists of irrelevant samples and errors, LI and LE, for each class, and first replace errors by irrelevant samples then replace the remaining irrelevant samples (if any) by other samples of Z_2 that have never been in Z_1.

Outliers degrade the project of any classifier. They will be usually identified as irrelevant prototypes, being moved from Z_1 to Z_2. Finally, Line 30 performs the training over the last set Z_1 to output the designed OPF classifier.

After learning, the classification of any sample $t \in Z_3$ is done by simply finding $s^* \in Z_1$ that satisfies Equation 2 and assigning label $L(s^*)$ as the class of t.

5. Results

We compare the OPF classifier with support vector machines (SVM [3]) using four databases with outliers and non-separable classes: Cone-torus from [12], Painted database (Figure 3(a)), MPEG-7 shape database [14], and WM/GM (white matter/gray matter) database [5]. The cone-torus database contains 400 samples and 3 non-separable classes while the painted database contains 5,867 samples with outliers and 4 classes. In both cases, the feature vectors are the sample (x, y) coordinates. The MPEG-7 database contains 1,400 2D shapes and 70 classes. To increase overlap (difficulty) between classes, we simply adopt the 126 most significant coeficients in the Fourier transform of the shapes as feature vector. The WM/GM database contains 1.5M voxels of WM and GM (2 classes) from MR-T1 images of phantoms with various levels of noise and inhomogeneity to produce outliers. The images and ground truth are available from [5], and the feature vector is the lowest and highest values around the voxel, and its intensity value. In all cases, function d is the Euclidean metric.



(a) (b)

Figure 3. (a) Painted database with outliers. (b) OPF learning curve on Z_2.

For all databases, we ramdomly selected the same percentage of samples from each class to create Z_1, Z_2 and Z_3. These percentages were 30% for Z_1, 30% for Z_2, and 40% for Z_3 in the first three databases. Only the WM/GM database used 0.1% for Z_1, 19.9% for Z_2 and 80% for Z_3.

For Z_1 and Z_2, we runned 10 iterations of Algorithm 2 to output the OPF classifier for test on Z_3. Figure 3(b) shows the learing curves for all databases. Note the usually non-decreasing behavior of the curves after the outliers be detected as irrelevant samples and moved to Z_2.

In SVM, we used a Gaussian kernel and computed support vectors for 10 new instances of Z_1 and Z_2 by ramdomly replacing samples between them, keeping the original proportions, and took the configuration with highest accuracy for test on Z_3.

The above learning and testing processes of SVM and OPF were also repeated for 10 distinct initial sets Z_1, Z_2, and Z_3 to compute mean and standard deviation of their accuracies over Z_3 (Table 1). OPF was usually more accurate and from 3 to 20 times faster than SVM.

Table 1. Mean and standard deviation of the accuracies for each database.

	OPF accuracy		SVM accuracy	
Database	mean	std. dev.	mean	std. dev.
Cone-torus	0.8757	0.0218	0.8147	0.0145
Painted	0.9838	0.0144	0.8763	0.0030
MPEG-7	0.6925	0.0049	0.5869	0.0088
WM/GM	0.9088	0.0006	0.9072	0.0009

6. Conclusions and future work

We use the IFT algorithm in sample feature spaces and propose pattern classifiers based on optimum path forests rooted at prototypes of training sets. The OPF classifier finds prototypes with none zero classification errors in the training sets and learns from errors in evaluation sets. Unseen test sets are used to assess OPF in comparision with SVM. From the learning curves of the OPF and its results on the test sets, we may conclude it is a robust classifier and usually more accurate than SVM for databases with outliers and non-separable classes.

We are currently evaluating OPF with other databases and its accuracy is usually higher than using SVM and ANN-MLP. Future works include to report these results and the extension of OPF to unsupervised classification.

References

[1] N. Arica and F. T. Y. Vural, *BAS: A Perceptual Shape Descriptor based on the Beam Angle Statistics*, Pattern Recognition Letters **24** (2003), no. 9-10, 1627–1639.

[2] A. Blum and S. Chawla, *Learning from Labeled and Unlabeled Data using Graph Mincuts.*, ICML '01: Proceedings of the 18rd international conference on Machine learning, 2001, pp. 19–26.

[3] B. E. Boser, I. M. Guyon, and V. N. Vapnik, *A training algorithm for optimal margin classifiers*, Proc. 5th Workshop on Computational Learning Theory, 1992, pp. 144–152.

[4] C. Chang and C. Lin, *LIBSVM: a Library for Support Vector Machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[5] D. Collins, A. Zijdenbos, V. Kollokian, J. Sled, N. Kabani, C. Holmes, and A. Evans, *Design and Construction of a Realistic Digital Brain Phantom*, IEEE Trans. on Medical Imaging **17** (1998), no. 3, 463–468. Available from: <`http://www.bic.mni.mcgill.ca/brainweb`>.

[6] R. Collobert and S. Bengio, *Links between perceptrons, MLPs and SVMs*, ICML '04: Proceedings of the twenty-first international conference on Machine learning, 2004, pp. 23.

[7] T. Cormen, C. Leiserson, and R. Rivest, *Introduction to Algorithms*, MIT, 1990.

[8] J. Cousty, G. Bertrand, L. Najman, and M. Couprie, *Watersheds, minimum spanning forests, and the drop of water principle* **IGM 2007-01** (2007).

[9] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed., Wiley-Interscience, 2000.

[10] A. X. Falcão, J. Stolfi, and R. A. Lotufo, *The image foresting transform: theory, algorithms, and applications*, IEEE Trans. on PAMI **26** (2004), no. 1, 19–29.

[11] L. J. Hubert, *Some applications of graph theory to clustering*, Psychometrika **39** (1974), no. 3, 283–309.

[12] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*, Wiley-Interscience, 2004.

[13] R. A. Lotufo and A. X. Falcão, *The ordered queue and the optimality of the watershed approaches*, Mathematical Morphology and its Applications to Image and Signal Processing, 2000, pp. 341–350.

[14] MPEG-7, *MPEG-7: The Generic Multimedia Content Description Standard, Part 1*, IEEE MultiMedia **09** (2002), no. 2, 78-87.

[15] N. Panda, E. Y. Chang, and G. Wu, *Concept boundary detection for speeding up SVMs*, ICML '06: Proceedings of the 23rd international conference on Machine learning, 2006, pp. 681–688.

[16] L. Reyzin and R. E. Schapire, *How boosting the margin can also boost classifier complexity*, ICML '06: Proceedings of the 23rd international conference on Machine learning, 2006, pp. 753–760.

[17] J. Shi and J. Malik, *Normalized cuts and image segmentation*, IEEE Trans. on Pattern Analysis and Machine Intelligence **22** (2000), no. 8, 888–905.

[18] B. Tang and D. Mazzoni, *Multiclass reduced-set support vector machines*, ICML '06: Proceedings of the 23rd international conference on Machine learning, 2006, pp. 921–928.

[19] C. T. Zahn, *Graph-Theoretical Methods for Detecting and Describing Gestalt Clusters*, IEEE Trans. on Computers **C-20** (1971), no. 1, 68–86.

Computing approximate geodesics and minimal surfaces using watershed and graph-cuts

Jean Stawiaski[1], Etienne Decencière[1] and François Bidault[2]

[1] *Centre de Morphologie Mathématique (CMM), Ecole des Mines de Paris, Fontainebleau, France*
`{Jean.Stawiaski,Etienne.Decenciere}@ensmp.fr`

[2] *Institut Gustave Roussy (IGR), Villejuif, France*
`bidault@igr.fr`

Abstract Geodesics and minimal surfaces are widely used for medical image segmentation. At least two different approaches are used to compute such segmentations. First, geodesic active contours use differential geometry to compute optimal contours minimizing a given Riemannian metric. Second, Boykov and Kolmogorov have proposed a method based on integral geometry to compute similar contours using a graph representation of the image and combinatorial optimization. In this paper we present a technique to compute approximate geodesics and minimal surfaces using a low-level segmentation and graph-cuts optimization. Our approach speeds-up the computation of minimal surfaces when a low-level segmentation is available.

Keywords: watershed, minimal surfaces, geodesics, graph-cuts.

1. Introduction

Computation of minimal surfaces and geodesics is a common problem in image processing [1, 2, 4, 6]. Like many other techniques, the segmentation by optimal surfaces is a classical minimization problem. At least two different approaches of the problem have been successfully applied to image segmentation: geodesic active contours [6] and graph-cuts segmentation [4]. The methods will be briefly introduced in Section 2 and Section 3.

In this paper we propose a method to compute approximate geodesics and minimal surfaces by using the watershed low-level segmentation (watershed from all the local minima of the image's gradient). Our approach is motivated by the simplification it offers in the formalization of the problem. We propose to compute a curve (or a surface) that minimizes a given geometric functional in the space of curves (or surfaces) composed by a sub-set of watershed contours. The segmentation is driven by the search of a minimal cut in a region adjacency graph. Experimental results show

G. Banon, et al. (eds),
Mathematical Morphology and its Applications to Signal and Image Processing, 349–360.
©2007 *MCT/INPE. Printed in Brazil.*
URL of this article: http://urlib.net/dpi.inpe.br/ismm@80/2007/03.19.13.04

that the approximation error is negligible on natural images. Results will be presented on 3D medical images.

2. State of the art

2.1 Geodesic active contours

The first application of differential geometry in image segmentation has been introduced by Kass et al. in [9]. The method, called "snakes", as well as many variants of active contours models, has been widely used for image segmentation. "Snakes" use a parametric representation of a curve. An important development has been introduced via a new representation of active contours [11, 13]. Parametric active contours have been replaced by an implicit representation of curves and surfaces via level-sets. This representation allows topological changes of curves and a better handling of numerical schemes to achieve the energy minimization. A further development of active contours has been introduced by Caselles et al. in [6] with "Geodesic active contours". This method simplifies the energy function to be minimized. The problem is formalized as the minimization of the energy:

$$E(C) = \int_0^{|C|_\varepsilon} g(||\nabla I(C(s))||)ds, \tag{1}$$

where $|C|_\varepsilon$ is the Euclidean length of a contour C, and s is the arc length on the contour. g is a positive and strictly decreasing function and ∇ is the gradient operator computed on the image I.

This method is equivalent to the minimization of the length of the curve C according to a Riemannian metric. The Riemannian metric depends here on the local gradient of the image I. For general curves, length in a Riemannian space can be written as:

$$|C|_R = \int_0^{|C|_\varepsilon} \sqrt{\tau_s^T D(C(s))\tau_s} ds, \tag{2}$$

where τ_s is a unit tangent vector to the contour C and D is a positive definite matrix, called the metric tensor, specifying the local Riemannian metric. In "Geodesic active contours" the local Riemannian metric is given by the following metric tensor:

$$D = \begin{pmatrix} g(\nabla I_x) & 0 \\ 0 & g(\nabla I_y) \end{pmatrix},$$

where $(\nabla I_x, \nabla I_y)$ are the components of the gradient of I.

"Geodesic active contours" minimize the Equation 1 via a gradient descent scheme and a level-sets representation of the curve. Unfortunately, the method is sensitive to initialization and the global minimum of Equation 1 is not always found. However the method can also be extended to three dimensions [7].

2.2 Cauchy-Crofton formulaes

Boykov and Kolmogorov [4] have considered the computation of minimal surfaces and geodesics based on the Cauchy-Crofton formulaes of integral geometry. Cauchy established a formula which relates the length of a curve C to a measure of a set of lines intersecting it. Let $L(\rho,\theta)$ be a straight line characterized in polar coordinates by the two parameters (ρ,θ). The Cauchy-Crofton formula establishes that the Euclidean length of a curve C is given by:

$$|C|_\varepsilon = \frac{1}{2}\int_0^\pi \int_{-\infty}^{\infty} N(\rho,\theta)d\rho d\theta, \tag{3}$$

where $N(\rho,\theta)$ is the number of intersections of $L(\rho,\theta)$ with C, and C is a regular curve. This formula can be extended to a Riemannian space, then the length of a curve C according to the metric tensor D is given by:

$$|C|_R = \frac{1}{2}\int_0^\pi \int_{-\infty}^{\infty} \frac{detD}{2(u_L^T D u_L)^{3/2}} N(\rho,\theta)d\rho d\theta, \tag{4}$$

where u_L is the unit vector in the direction of line L. This formula is verified by any continuously differentiable and regular curve in $\mathbb{R}^2$ [12].

Let N_g be a neighborhood system on a discrete grid. N_g can be described by a finite set of undirected vectors e_k, $N_g = \{e_k : 1 < k < n_g\}$. Each vector e_k generates a family of lines as shown in Figure 1. Each line in a family is separated by a distance $\Delta\rho_k$ from the closest line of the family. Now let θ_k be a discrete angular parameter. For a fixed θ_k, we obtain a family of parallel lines separated by a distance $\Delta\rho_k$ as shown Figure 1.

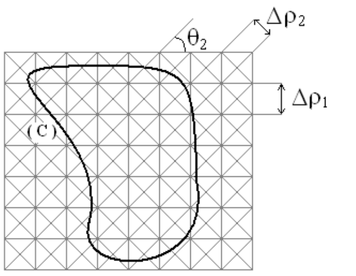


Figure 1. 8-Neighborhood system. Cauchy-Crofton formula established a link between a finite set of lines and the Euclidean length of a curve C.

The discretization of Equation 3 gives the following approximation of the Euclidean length of the curve C:

$$|C|_\varepsilon \approx \frac{1}{2}\sum_{k=1}^{n_g}(\sum_i n_c(i,k)\Delta\rho_k)\Delta\theta_k = \sum_{k=1}^{n_g} n_c(k)\frac{\delta^2\Delta\theta_k}{2|e_k|}, \tag{5}$$

where i indexes the k^{th} family of lines. $n_c(i,k)$ counts the number of intersections of line i of the k^{th} family of lines with the curve C. $n_c(k) = \sum_i n_c(i,k)$ is the total number of intersections of the k^{th} family of lines with C.

3. Graph-cuts

3.1 Basics

Graph-cuts are based on the well-known combinatorial problem of finding a minimal cost cut in a weighted graph. Suppose that each arc (i,j) of a graph G has assigned to it a non negative number $c(i,j)$ called the capacity or the weight of the arc. This capacity is seen as the maximum amount of some commodity that can "flow" through the arc. Let us consider the problem of finding a maximal flow from a node s, called the source, to a node t, called the sink. Finding the maximal possible flow between s and t is related to finding a minimal cut in the graph. A (s-t) cut is identified by a pair (S,T) of complementary subsets of nodes, with $s \in S$ and $t \in T$. The cost of the cut is defined by:

$$c(S,T) = \sum_{i \in S} \sum_{j \in T} c(i,j). \tag{6}$$

The minimal cut can be efficiently computed in polynomial time using classical combinatorial algorithm such as the Ford-Fulkerson algorithm or more efficient algorithms as the one proposed by Boykov et al. in [5]. Graph cuts are well suited for image segmentation since a node can represent a pixel and edges represent neighborhood relations between pixels. Graph-cuts have already been used in many imaging applications [3, 4, 10].

3.2 Computing geodesics and minimal surfaces via graph-cuts

Boykov and Kolmogorov have considered the computation of minimal surfaces and geodesics with interactive placement of markers. The user has to specify "background" and "object" seeds and their method finds automatically the optimal curve (or surface) separating the two sets of seeds. The image is represented by a graph. Two additional nodes s and t are respectively connected to "object" seeds and "background" seeds. "s-links" and "t-links", arcs connected to s or t, have infinite capacity to ensure that the sets S and T respectively contain a "background" seed and a "foreground" seed.

The aim of the method is to relate the cost of a graph-cut to the length of a underlying curve as shown in Figure 1. Let us consider an image embedded on a discrete grid and let N_g be a neighborhood system on the image. As described in the previous section, the neighborhood system defines a family of lines. The cost of a (s-t) cut in the constructed graph is then equal to:

$$c(S,T) = \sum_{i \in S} \sum_{j \in T} c(i,j) = \sum_{k=1}^{n_g} n_c(k) w_k, \tag{7}$$

where $n_c(k)$ is the number of arcs of family k that connect S to T, and w_k is the weight of the arcs of family k.

The Cauchy-Crofton formula given by Equation 5 can be directly used to set arcs weights such that the cost of a graph-cut approximates the Euclidean length of the contour separating the two sets S and T:

$$c(S,T) = \sum_{k=1}^{n_g} n_c(k) w_k \quad with \quad w_k = \frac{\delta^2 \Delta\theta_k}{2|e_k|}. \tag{8}$$

The previous relation can also be extended to deal with Riemannian metric by using the following weights:

$$w_k(p) = \frac{\delta^2.|e_k|^2.\Delta\theta_k.det(D(p))}{2(e_k^T D(p) e_k)^{3/2}}, \tag{9}$$

where $w_k(p)$ is the weight of the arcs leaving the node p, and $D(p)$ is the local Riemannian metric at point p. The previous expressions can also be extended to 3D spaces [4].

These formulaes show explicitly that the cost of a graph-cut is related to the geometric length of the contour separating the sets S and T defined by the cut. Unfortunately existing methods are computationally costly and cannot always be used interactively on large datasets such as 3D medical images. Inspired by the approaches presented in Section 2 and Section 3, we propose a method to compute fast approximate geodesics and minimal surfaces from an initial low level segmentation of the image.

4. Approximate geodesics and minimal surfaces using watershed segmentation

4.1 Problem statement

The combination of graph-cuts with a watershed low-level segmentation (watershed from all the local minima of the image's gradient) provides us an explicit way to compute geodesics and minimal surfaces. Our basic assumption is that the geodesic to be computed is embedded in the watershed low-level segmentation. This proposition is motivated by two observations. Firstly, the watershed transform (computed from the local minima of image's gradient), without pre-processing or marker selection, produces an over-segmentation of real images. Secondly, the watershed lines contain all major boundaries of real images. Thus, we propose to solve the following combinatorial problem: finding a curve composed of a finite union of watershed lines such that the curve minimizes a given geometric functional. We will solve this problem by using graph-cuts optimization on a region adjacency graph, as suggested by Li et al. [10].

4.2 Combining graph-cuts and watershed segmentation

Following [6], we will consider a geodesic curve C than can be computed via the minimization of the energy given by Equation 1. Let us consider the graph $G = [X, U, W]$ of the watershed regions where $X = \{x_k\}$ is the set of nodes (i.e the regions of the watershed transform), U is the set of arcs (i.e the neighborhood relations between regions) and W is the weights of the arcs as illustrated in Figure 2.

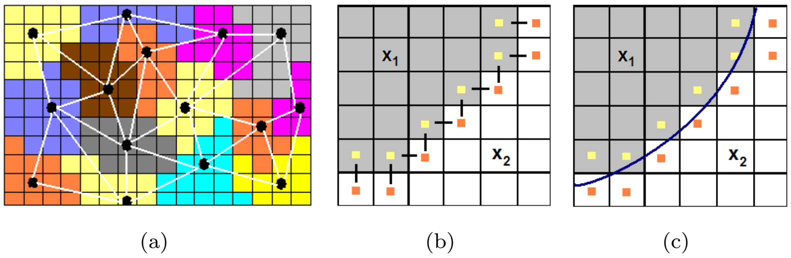


Figure 2. (a) Region adjacency graph of a low-level watershed segmentation. (b) The sets of pixels considered to compute boundary properties between adjacent regions, with a V_4 adjacency relation. (c) An implicit curve defined by the regions x_1 and x_2.

We present a way of defining arcs weight such that a cut partitions the image by an approximate minimal curve (curve of minimal length in a Riemannian space). Let us define $F_{(x_i,x_j)}$ as the border between two regions x_i and x_j of the low-level watershed segmentation:

$$F_{(x_i,x_j)} = \{(p_m, p_n) \mid p_m \in x_i,\ p_n \in x_j,\ (p_m, p_n) \in N\}. \tag{10}$$

One should note that the set $F_{(x_i,x_j)}$ depends on the adjacency relation N. This set of nodes implicitly describes a set of curves between the regions x_i and x_j as illustrated in Figure 2(c). Let us define $C_{(x_i,x_j)}$ as the set of curves that can go through the nodes of $F_{(x_i,x_j)}$. Thus we can explicitly compute the energy $E(C_{(x_i,x_j)})$ for all pairs of regions using the Cauchy-Crofton formulaes detailed in Section 2. Note also that if regions x_i and x_j are not adjacent, $F_{(x_i,x_j)}$ and $C_{(x_i,x_j)}$ are empty sets.

Let us define a strictly positive function g of $(F_{(x_i,x_j)})$ as:

$$g(F_{(x_i,x_j)}) = \sum_{(p_m,p_n)\in F_{(x_i,x_j)}} \frac{1}{1+||I(p_m)-I(p_n)||^2}, \tag{11}$$

$$g(\emptyset) = 0. \tag{12}$$

The function g works as an edge indicator of the image I and takes a small value if the gradient of I is high between the regions x_i and x_j. One should note that the function g approximates the Riemannian length of the implicit curves $C_{(x_i,x_j)}$ in case of the 4-neighborhood system. According to the Cauchy-Crofton Formulaes, the number of adjacent pixels ($(p_m, p_n) \in F_{(x_i,x_j)}$) indicates the number of intersection of an implicit curve between the regions x_i and x_j with the horizontal and vertical lines describing the 4-neighborhood system:

$$|C_{(x_i,x_j)}|_R = E(C_{(x_i,x_j)}) \approx g(F_{(x_i,x_j)}). \tag{13}$$

Alternatively, the Cauchy-Crofton formulaes can also be used to compute the approximate Riemannian length of the curves $C_{(x_i,x_j)}$ between two adjacent regions in case of a larger neighborhood system. However in this section we will only consider the 4-neighborhood system for simplicity.

The cost of a (s-t) cut in the region adjacency graph weighted by the function g is equal to:

$$c(S,T) = \sum_{x_i\in S}\sum_{x_j\in T} w(x_i,x_j), \tag{14}$$

$$c(S,T) = \sum_{x_i\in S}\sum_{x_j\in T} (g(F_{(x_i,x_j)})). \tag{15}$$

As a consequence a (s-t) cut in the region adjacency graph is equal to the Riemannian length of an curve between the sets S and T. Considering the weighting function given by Equation 11, the minimal cut of the weighted adjacency graph of watershed regions is equal to:

$$min_{(S,T)} c(S,T) = min_{(C\in(\bigcup C_{(x_i,x_j)}))} E(C), \tag{16}$$

where $(\bigcup C_{(x_i,x_j)})$ is the union of all the implicit curves defined by the watershed regions.

Our minimization problem is reduced to the search of a curve among all curves implicitly described by the watershed regions instead of searching among all curves in the domain of the image I. This approximation reduces drastically the search space.

4.3 Computing minimal surfaces

The method can easily be extended to three dimensions by considering integrals on surfaces instead of curves. The aim of the segmentation task is now to find a surface S that minimizes the following energy:

$$E(S) = \int\int_S g(||\nabla I(x,y)||)dxdy, \tag{17}$$

where S is a surface and g a positive and strictly decreasing function.

Thus we can use the same capacities defined in Equation 11, and apply it on the region adjacency graph in 3D:

$$w(x_i, x_j) = g(F_{(x_i,x_j)}). \tag{18}$$

In 3D, $F_{(x_i,x_j)}$ defines implicitly a set of surfaces $S_{(x_i,x_j)}$ between the regions x_i and x_j. Thus the minimal cut of the region adjacency graph in 3D is equal to:

$$min_{(S,T)}c(S,T) = min_{(S\in(\bigcup S_{(x_i,x_j)}))}E(S). \tag{19}$$

4.4 Adding geometric constraints

Using a region adjacency graph instead of the pixel adjacency graph can be advantageous in some situations. A wide class of geometric functionals can be computed on each regions of the watershed transform. As a consequence, a large class of geometric functionals can be added to the energy defined by Equation 1. For instance, it remains unclear how to introduce curvature constraints in the graph-cuts method used at the pixel level, but it is clear that a curvature term can easily be added in our methodology. For instance curvature of the border between two adjacent regions can be computed and used to add a shape constraint to the energy to be minimized.

5. Results

This section presents some results obtained by our method on 3D medical images. Figure 3 illustrates our segmentation method (Figure 3(b)) and compares it with the classical marker-controlled watershed segmentation (Figure 3(d)) and the minimal surface computed with the technique proposed by Boykov et al. in [4] (Figure 3(c)). Our method outperforms the marker-controlled segmentation and produces approximately the same segmentation as the graph-cuts method proposed by Boykov et al. in [4].

The next example illustrates the method on a 3D CT image. Figure 4 illustrates the segmentation of a liver. The liver presents low-contrasted boundaries and the segmentation of such organs remains a difficult task.

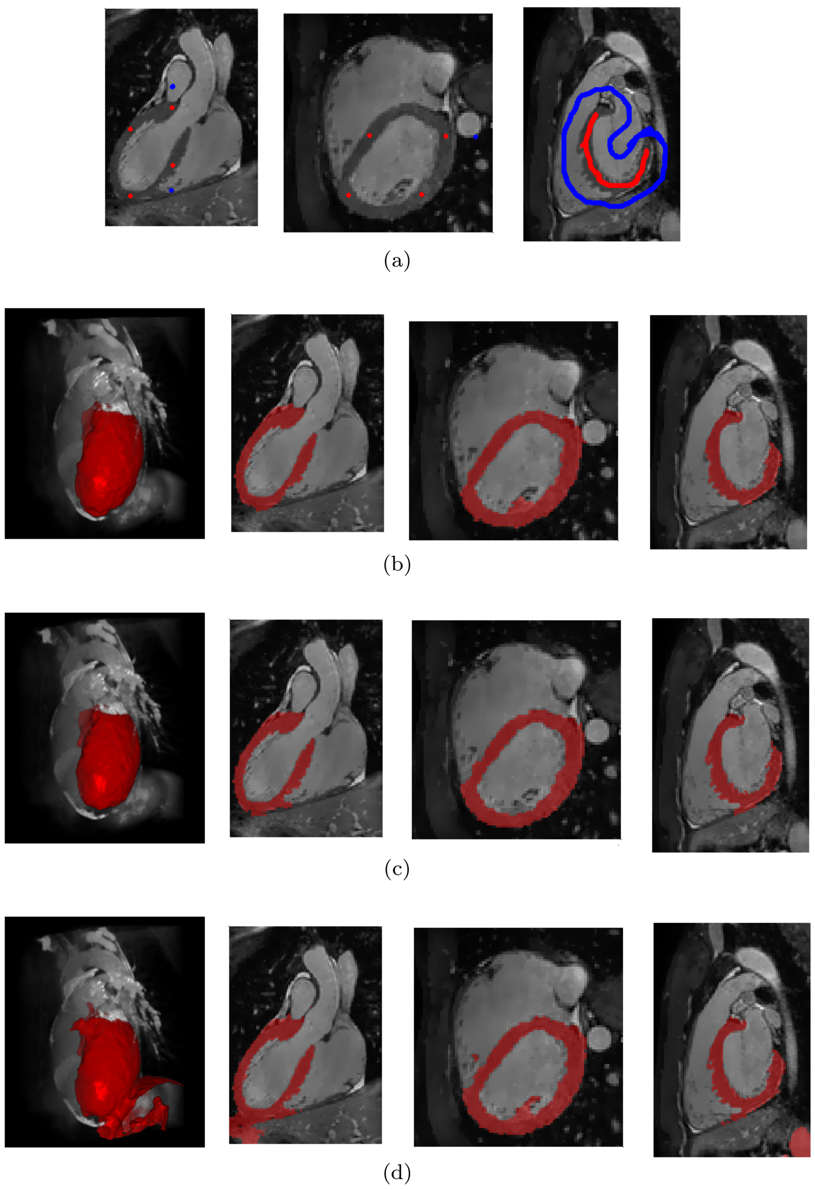


Figure 3. (a) Heart MRI superposed with user-provided markers. (b) Approximate minimal surface by our method. (c) Graph cuts minimal surface. (d) Marker-controlled watershed segmentation.

For this application minimal surface remains a leading method. The graph-cuts used on the watershed adjacency graph (Figure 4(b)) improves the results given by the classical marker-controlled watershed segmentation (Figure 4(d)) and speeds up the graph-cuts method proposed by Boykov et al. [4].

Extensive tests have been undertaken with 3D medical images provided by the Institut Gustave Roussy[1] (mainly CT images of the thorax) and the Centre for Advanced Visualization and Interaction[2] (MRI images of the heart). According to specialists, the results are very promising, even for the interactive segmentation of anatomical structures which are difficult to contour, such as the liver.

Table 1 illustrates the computation time needed by the three methods we considered: the marker-controlled watershed, our method, and the minimal surfaces by graph-cuts proposed in [4]. The results shows that our method reduces drastically the computation time needed by graph based techniques. Moreover our method do not seem to affect the results quality.

Table 1. Comparisons of computation time. (Laptop Pentium Core Duo 2.16 Ghz, 1Go memory)

Image	*Watershed*	*Our method*	*Graph cuts*
Heart MRI	1,5 sec.	4,2 sec.	15,2 sec.
Liver CT	9,7 sec.	41,6 sec.	1400,2 sec.

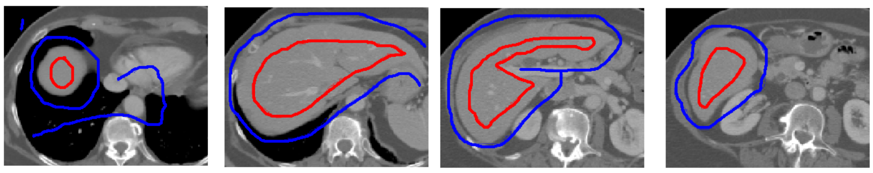
5.1 Conclusion

Considering that the watershed transform contains all major boundaries in real images, our approximate segmentation is in practise quite efficient. However the graph-cuts approach works slower than the classical marker-controlled watershed but offers more stable results. In the other hand our method is not as precise as the graph-cuts method proposed by Boykov et al. [4], but it offers a good trade off between speed and precision.

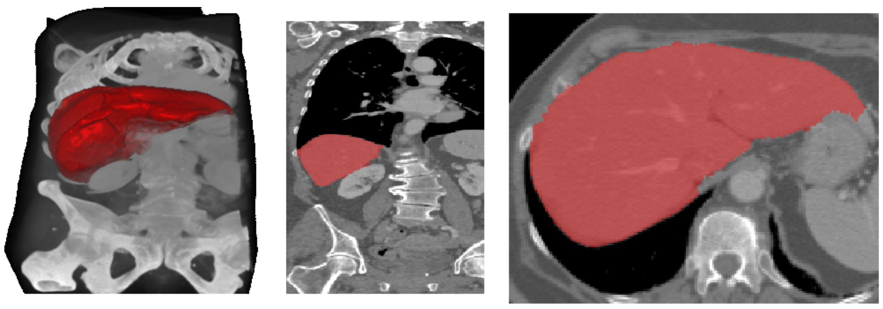
A Graph-cuts approach cannot always be used on large images when the graph considered is the pixel adjacency graph because of the memory requirements and the computational complexity of the method. The developed method can efficiently be used on large images considering the region adjacency graph instead of the pixels graph. Our method do not seem to introduce large biases in the resulting segmentation of natural images. Limitations of our methods are quite clear since it can only be used when an

[1]The Institut Gustave-Roussy is a non-profit private institution, exclusively devoted to oncology, located near Paris, France

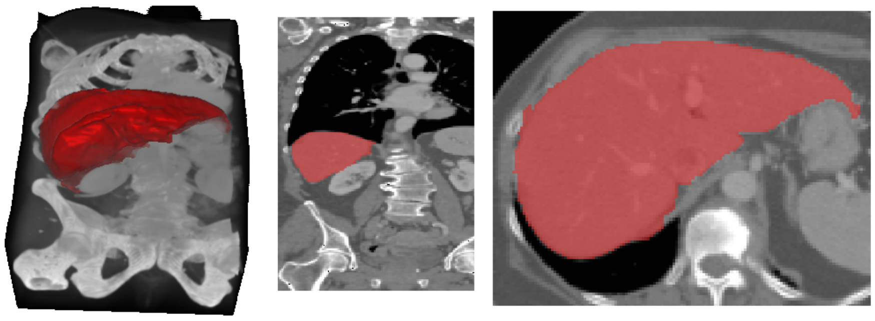
[2]University of Aarhus, Denmark



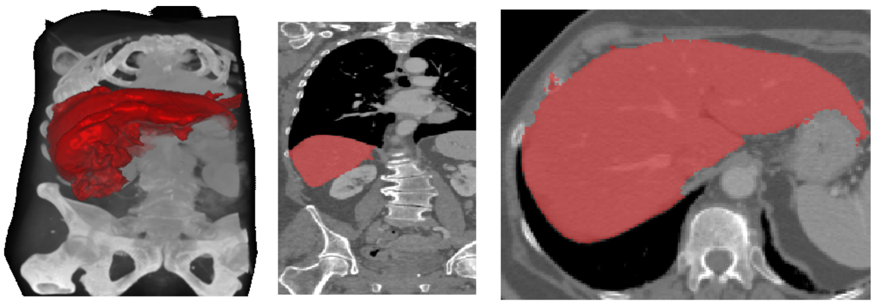
(a)



(b)



(c)



(d)

Figure 4. 3D CT image . (a) User-provided markers. (b) Approximate minimal surface by our method. (c) Graph-cuts minimal surface. (d) Marker-controlled watershed segmentation.

over-segmentation can be obtained. Graph-cuts can also be used on other adjacency graphs. For instance λ-flat zones [8] adjacency graph should be a good solution to increase the precision of our method since it can offer a pixel-precision in some situations.

Our methodology can take into account a wide class of geometric functionals since we can compute all kind of measures on the boundaries of the watershed regions. For instance this method can take into account curvature of the boundaries, which remains impracticable for classical graph-cuts methods used at the pixel level.

6. Acknowledgements

We would like to thank Région Ile-de-France and the Cancéropôle Ile-de-France for funding our research on medical image segmentation.

References

[1] B. Appleton and H. Talbot, *Globally optimal geodesic active contours*, JMIV **23** (2005), 67–86.

[2] ———, *Globally Minimal Surfaces by Continuous Maximal Flows*, PAMI **28(1)** (2006), 106–118.

[3] Y. Boykov and M. P. Jolly, *Interactive Graph Cuts for Optimal Boundary and Region Segmentation of Objects in N-D Images*, ICCV (2001), 105–112.

[4] Y. Boykov and V. Kolmogorov, *Computing Geodesics and Minimal Surfaces via Graph Cuts*, ICCV (2003), 26–33.

[5] ———, *An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision*, IEEE Transactions on Pattern Analysis and Machine Intelligence **26** (2004), 1124–1137.

[6] V. Caselles, R. Kimmel, and G. Sapiro, *Geodesic Active Contours*, International Journal of Computer Vision (1995), 694–699.

[7] V. Caselles, R. Kimmel, G. Sapiro, and C. Sbert, *Minimal surfaces: A three dimensional segmentation approach*, Technion EE Pub. 973, (1995).

[8] J. Crespo, R. Schafer, J. Serra, C. Gratin, and F. Meyer, *The flat zone approach : a general low-level region merging segmentation method*, Signal Processing **62** (1997), 37–60.

[9] M. Kass, A. Witkin, and D. Terzopoulos, *Snakes: Active contour models*, International Journal of Computer Vision **1(4)** (1988), 321–331.

[10] Y. Li, J. Sun, C. Tang, and H. Shum, *Lazy snapping*, IGGRAPH, ACM Transaction on Graphics **23** (2004), 303–308.

[11] S. Osher and R. Fedkiw, *Level set methods and dynamic implicit surfaces*, Springer, 2003.

[12] L. A. Santalo and R. Fedkiw, *Integral Geometry and Geometric Probability*, Addison-Wesley, 1979.

[13] J. A. Sethian, *Level Set Methods and Fast Marching Methods*, Cambridge University Press.

VII

TEXTURE AND GEOMETRICAL SEGMENTATION

Morphological texture gradients: Definition and application to colour and texture watershed segmentation

Jesús Angulo

Centre de Morphologie Mathématique (CMM), Ecole des Mines de Paris, Fontainebleau, France
`jesus.angulo@ensmp.fr`

Abstract This paper deals with a morphological approach to calculate texture gradients and it is shown how to use them for image segmentation according to the texture; and more generally, for joint colour/texture segmentation (i.e., structural segmentation). The starting point is a decomposition of the colour image into two components: the object layer and the texture layer. A multi-scale local analysis from the texture layer is built using morphological operators (openings/closings or levelings) to define the gradients of texture. The proposed texture gradient is then combined with the colour gradient to produce mixed segmentations by watershed.

Keywords: granulometry, leveling, colour/texture decomposition, colour gradient, texture gradient, watershed transform.

1. Introduction

The classical paradigm of morphological segmentation is the watershed transformation with imposed markers [14], which is one of the most powerful segmentation techniques. Watershed-based hierarchical approaches allow addressing fields where markers cannot be easily defined (e.g., natural images, video-surveillance, etc.). Two main hierarchical techniques can be distinguished: 1) non-parametric waterfalls algorithm [4], which eliminates the contours completely surrounded by stronger contours; and 2) hierarchies based on extinction values [12, 20], which allow to select the minima used in the watershed according to different morphological criteria (in particular, the volume, which combines the size and the contrast of the regions, defines a good criterion to evaluate the visual relevance of regions). These algorithms are built on a scalar gradient. A colour gradient must be calculated to apply the watershed on a colour image. According to our previous works [2], we propose to compute a complete colour gradient in a luminance/saturation/hue representation, which is relatively robust towards illumination condition variations. More precisely, if

$\mathbf{f}(x) = (f_L(x), f_S(x), f_H(x))$ denotes a colour image in the LSH representation, its colour gradient is given by:

$$\varrho_{col}(\mathbf{f})(x) = (1 - f_S(x)) \times \varrho(f_L)(x) + f_S(x) \times \varrho^{\circ}(f_H)(x) + \varrho(f_S)(x),$$

where $\varrho(g)(x)$ is the morphological gradient of a scalar function $g(x)$ (in this case, the luminance $f_L(x)$ and the saturation $f_S(x)$) and $\varrho^{\circ}(a)(x)$ is the circular centered gradient of an angular function $a(x)$ (in this case, the hue component $f_H(x)$).

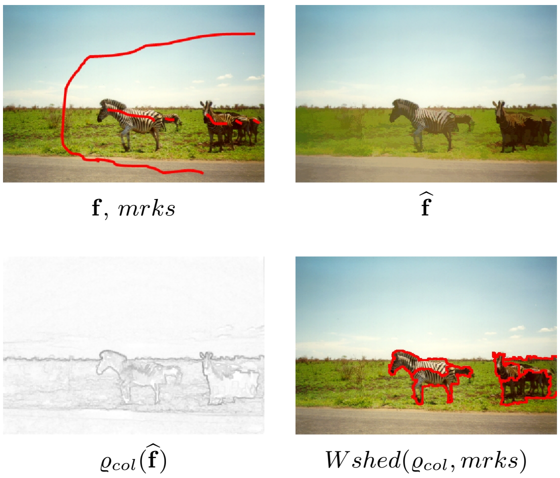

$\mathbf{f}$, $mrks$
$\widehat{\mathbf{f}}$
$\varrho_{col}(\widehat{\mathbf{f}})$
$Wshed(\varrho_{col}, mrks)$

Figure 1. Example of colour segmentation by markers-based watershed.

In the traditional way to segment an image by watershed transformation, the colour image is previously filtered by means of a connected operator, typically a leveling [13], $\lambda(m, f)$ (f is the reference image and $m(f)$ is the marker image, which is a rough simplification of the reference image), which simplifies textures and eliminates small details, but preserving the contours of remaining objects. For colour images, a marginal leveling can be applied for each component R, G, B or a total colour leveling can be calculated [1]. In any case, the leveling needs an image marker which determines the structures to be preserved, i.e.,

$$\widehat{\mathbf{f}} = \lambda(ASF_{nB}(\mathbf{f}), \mathbf{f}),$$

where ASF_{nB} is an alternate sequential filter of size n and B is an isotropic structuring element (other filters such as the Gaussian filters can be used to build the marker). Then, the watershed is calculated on the colour gradient of $\widehat{\mathbf{f}}$. The example of Figure 1 illustrates the segmentation with a marker for each object of interest (each zebra and an additional marker for the background). As we can observe, the colour information does not make possible to extract correctly the object contours.

Indeed, the texture is in certain images a very discriminating information for object separation. However, to introduce texture into the segmentation is not so simple as for the colour: texture is a regional notion which is difficult to quantify. In [6], Hill et al. proposed a method to build a texture gradient starting from a wavelet transformation, which is then used with the watershed to segment grey-level images. The combined use of colour and texture is the topic of a certain number of recent works. Ma and Manjunath [8] introduced the interest of Gabor filters for texture image segmentation. Vanhamel et al. introduced in [21] a marginal approach to apply Gabor filters to each component of a colour image and thus to construct a colour/texture feature space for segmentation. In a similar way, Hoang et al. [7] used Gabor filters to measure colour/texture and the segmentation is obtained by k-means. The works by Malik et al. [10] are also based on banks of Gaussian filters to calculate a texture gradient which is then combined with luminance and colour gradients in a supervised learning framework. Sofou et al. [18] introduced a joint intensity/texture segmentation by a PDE-based watershed, where texture is measured by a demodulation filter bank. This last work starts from an image decomposition according to the model $f = u + v$ by Y. Meyer [15], where u is the "cartoon component" (homogeneous zones of the objects) and v is the "texture oscillation". This model was initially studied within the framework of a variational approach by Vese and Osher [22]. More recent works, for example Patwardhan and Sapiro [16] and Aujol et al. [3], explore fast variational algorithms for the calculation of the images u and v. Sofou et al. proposed to obtain the texture component as the residue of a leveling, i.e., $v = f - u = f - \lambda(m, f)$ (the marker m is a Gaussian filter of f).

In this paper, we focus on a similar framework to that of Sofou et al. [18], but less expensive in computational terms. Our starting point is also a colour image decomposition of $\mathbf{f}$ into two components:

$$\mathbf{f} \triangleq \widehat{\mathbf{f}} \uplus f_{tex},$$

where $\widehat{\mathbf{f}}$ is the *object layer* and f_{tex} is the *texture layer*. The texture layer is obtained as the residue of the components of luminance, i.e., $f_{tex} = f_L - \widehat{f_L}$, because the texture variations are mainly associated to the luminance.

2. Granulometries and morphological multi-scale analysis

A granulometry is the study of the size distribution of the objects of an image [11, 17]. Formally, for the discrete case, a granulometry is a family of openings $\Gamma = (\gamma_n)_{n \geq 0}$ that depends on a positive parameter n (which expresses a size factor) such as: 1) $\gamma_0(f) = f$; 2) $f \leq g \Rightarrow \gamma_n(f) \leq \gamma_n(g), \forall n \geq$

0, $\forall f, g$; 3) $\gamma_n(f) \leq f, \forall n \geq 0$, $\forall f$; 4) and γ_n verifies the semi-group absorption law; i.e., $\forall n, m \geq 0$, $\gamma_n\gamma_m = \gamma_m\gamma_n = \gamma_{\max(n,m)}$. Moreover, a granulometry by closings (or anti-granulometry) can be defined as a family of increasing closings $\Phi = (\varphi_n)_{n\geq 0}$. In practice, the most useful granulometry and anti-granulometry are those associated to morphological openings/closings: $\gamma_n(f) = \delta_{nB}(\varepsilon_{nB}(f))$ and $\varphi_n(f) = \varepsilon_{nB}(\delta_{nB}(f))$ respectively, where B is a structuring element of unit size (typically a disc or a segment of straight line) and $n = 1, 2, \cdots$. The greedy algorithms for granulometries involve consequently openings (closings) of increasing size, and thus they are relatively expensive. However, optimised fast algorithms for granulometry computation have been developed by Vincent [23].

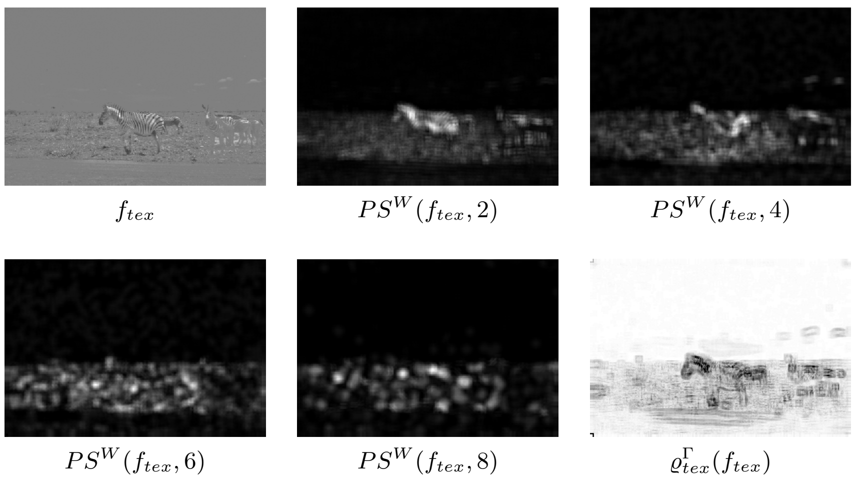


Figure 2. Texture layer, local granulometry (window $W_x = 10 \times 10$) by isotropic openings, morphological texture gradient.

The granulometric analysis of an image f with respect to Γ consists in evaluating each opening of size n with a measurement: $\mathcal{M}(\gamma_n(f))$ (where $\mathcal{M}$ is the integral of scalar function values). The granulometric curve, or pattern spectrum [9], of f with respect to Γ and Φ, $PS_{\Gamma,\Phi}(f, n)$ or $PS(f, n)$, is defined by the following normalised mapping:

$$PS(f,n) = \frac{1}{\mathcal{M}(f)} \begin{cases} \mathcal{M}(\gamma_n(f)) - \mathcal{M}(\gamma_{n+1}(f)), & for\ n \geq 0, \\ \mathcal{M}(\varphi_{|n|}(f)) - \mathcal{M}(\varphi_{|n|-1}(f)), & for\ n \leq -1. \end{cases}$$

The value of pattern spectrum $PS(f, n)$ for each size n corresponds to a measurement of bright structures of size n (similarly, the dark structures are obtained by closings). The pattern spectrum $PS(f, n)$ is a probability density function (i.e., a histogram): a peak or mode in PS at a given scale n indicates the presence of many image structures of this scale or size.

Granulometric size distributions can be used as descriptors for texture classification. However, the texture descriptor $PS(f,n)$ is global to the image f, and if f contains more than one texture, the classification should be carried out at pixel level. This is the concept behind the granulometric local analysis [5], which consists in calculating a local pattern spectrum, or more precisely a pattern spectrum in a window $W_x = size_h \times size_v$ ($size_h$ is the horizontal size in pixels and $size_v$ the vertical one) centered at pixel x. The local pattern spectrum $PS^{W_x}(f,n)$, or simply $PS^W(f,n)$, is obtained by computing the function $PS(f_{W_x}, n)$ for each pixel x, where f_{W_x} is the restriction of the image f to the window W_x. This method is very expensive from a computational viewpoint. A faster approach to obtain $PS^W(f,n)$ is based on the computation of only one series of openings/closings and then, for each pixel x, to calculate locally the integral in W_x, i.e., $\mathcal{M}^{W_x}(g) = \sum_{y \in W_x} g(y)$. As result of this computation, a granulometric curve is obtained for each pixel. This local texture descriptor can be used to classify the various zones of texture in an image [5].

In our case, this local granulometric analysis must be done on the texture layer f_{tex} and the series of images which code this analysis is denoted by $\{t_k^{\Gamma\Phi}(x)\}_{k\in K} = \mathbf{t}^{\Gamma\Phi}(x)$, where

$$t_k^{\Gamma\Phi}(x) = PS^{W_x}(f_{tex}, k).$$

The function $t_k^{\Gamma\Phi}(x)$ is named the image of local energy of size k ($k \geq 0$ for the bright structures and $k \leq -1$ for the dark structures). In Figure 2 the texture layer for the image of zebras is shown, as well as the images of local energy associated to the local granulometry by isotropic openings (window $W_x = 10 \times 10$ and $K = \{-16, -14, \cdots, -2, 2, 4, \cdots, 16\}$). It is observed that the structures of f_{tex} have high values of local energy for their corresponding sizes. In the example, only four images $t_k^{\Gamma}(x)$ are shown (bright structures); a dual analysis $t_k^{\Phi}(x)$ provides the local energies for the different scales of dark structures. The choice of the size of the window depends on the "texture scales". However, its influence is limited: for all the examples of natural images presented in this study the choice $W_x = 10 \times 10$ showed to be appropriate. In Figure 3 another example of colour/texture decomposition is given, including also two images of local energy. Obviously we can use other non isotropic structuring elements B in order to describe, for example, orientated textures.

In mathematical morphology, we can build other multi-scale analysis using other operators different from openings/closings. Let $ASF_n(f) = \varphi_n\gamma_n \cdots \varphi_2\gamma_2\ \varphi_1\gamma_1(f)$ be the alternate sequential filter of size n (we can define another family of filters by reversing the order of the opening/closing). The family $\Xi = (ASF_n)_{n\geq 0}$ verifies the semi-group law of absorption and consequently, it allows to define multi-scale simplification (or selection, by considering the residues) of the structures of f_{tex}. In addition, if each scale is associated to a leveling, the new family of transformations, $\Lambda = (\lambda_n)_{n\geq 0}$

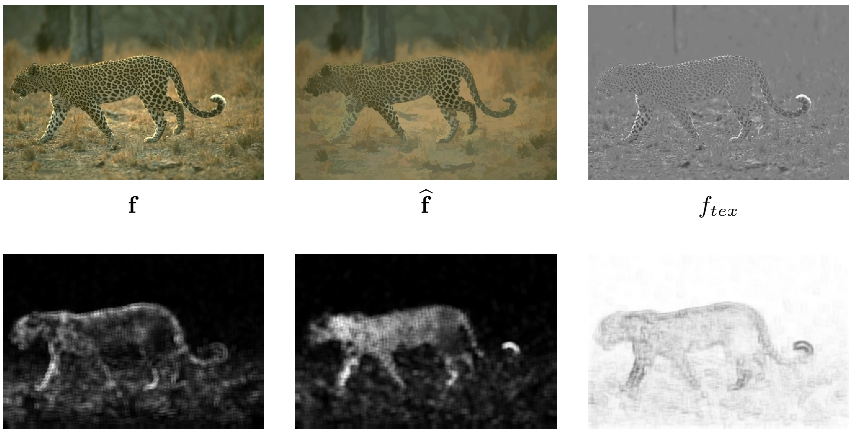


Figure 3. Colour/texture decomposition, two images of local energy, morphological texture gradient.

such that $\lambda_n(f) = \lambda(ASF_n(f), f)$, provides a decomposition of the reconstructed objects according to each scale n. It should be noted that using the levelings, both bright/dark objects of size n appear in the same image.

This leveling-based quantitative analysis of the objects associated to each size n makes it possible to define a pseudo-granulometric curve, named Λ-pattern spectrum, which is defined as follows:

$$\Lambda PS(f,n) = \mathcal{M}(\lambda_n(f)) - \mathcal{M}(\lambda_{n-1}(f)),$$

for $n \geq 0$. As for the granulometry, a local version of $\Lambda PS(f,n)$ is defined by computing the measure in a window W centered in each pixel. The associated series of images of local energy, i.e.,

$$t_k^{\Lambda}(x) = \Lambda PS^{W_x}(f_{tex}, k),$$

gives an alternative multi-scale representation (typically $k \in K = \{2, 4, \cdots, 16\}$). It must be remarked that for $t_k^{\Gamma\Phi}$ and t_k^{Λ} the maximal size k is limited by the size of leveling used to build $\widehat{\mathbf{f}}$, e.g., the maximal size of structures in f_{tex}. Other morphological multi-scale decompositions could be used in order to define other texture descriptors: operators associated with dynamics, area, volume, etc. [19, 20].

3. Morphological gradients of texture

We consider now the alternatives to calculate a gradient, associated to the multi-scale analysis, which allows determining the contours for the zones of different textures.

In each point x, the morphological gradient $\varrho(x)$ of unit size $B(x)$ of an image g can be written in terms of increments, i.e., $\varrho(g)(x) = \delta_B(g)(x) - \varepsilon_B(g)(x) = \vee[g(x) - g(y), y \in B(x)]$. Using this formulation, it is possible to use an Euclidean distance to define a gradient of morphological type for the series of images of local energy, i.e.,

$$\varrho_{tex}(f_{tex})(x) = \vee_y[d_E(\mathbf{t}(x), \mathbf{t}(y)), y \in B(x)],$$

where $d_E(\mathbf{t}(x), \mathbf{t}(y)) = \sqrt{\sum_{k \in K}(t_k(x) - t_k(y))^2}$ is the Euclidean distance between the two pixels x and y for all the images of local energy.

Besides this vectorial gradient, it is also possible to define another kind of gradient, by combining the gradients of each scalar image of energy. Various tests showed that the gradient by supremum, i.e.,

$$\varrho_{tex}(f_{tex})(x) = \bigvee_{k \in K} [\varrho(t_k(x))],$$

is as useful for the segmentation as the vectorial gradient defined by Euclidean distance, and easier to compute. Figure 2 gives also the morphological gradient $\varrho^{\Gamma}_{tex}(f_{tex})$ calculated according to the sup of scalar gradients. For the example of Figure 3, the gradient has been computed using the vectorial formulation.

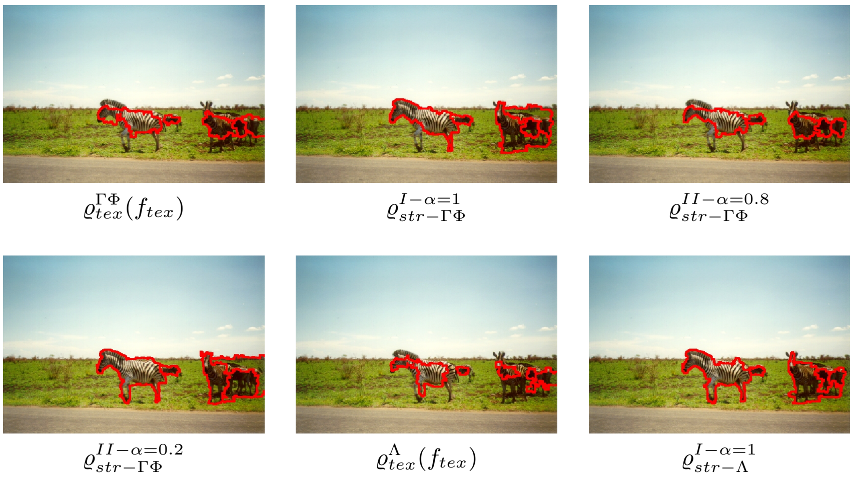

$\varrho^{\Gamma\Phi}_{tex}(f_{tex})$
$\varrho^{I-\alpha=1}_{str-\Gamma\Phi}$
$\varrho^{II-\alpha=0.8}_{str-\Gamma\Phi}$
$\varrho^{II-\alpha=0.2}_{str-\Gamma\Phi}$
$\varrho^{\Lambda}_{tex}(f_{tex})$
$\varrho^{I-\alpha=1}_{str-\Lambda}$

Figure 4. Examples of markers-based watershed segmentation with texture gradients and structural gradients (colour+texture), i.e., $Wshed(\varrho, mrks)$.

These texture gradients, derived from the images of local energy $\{t_k^{\Gamma\Phi}(x)\}$ and $\{t_k^{\Lambda}(x)\}$, respectively $\varrho^{\Gamma\Phi}_{tex}(x)$ and $\varrho^{\Lambda}_{tex}(x)$, can be used with the watershed to segment the image into regions according to the texture. See the two corresponding results in Figure 4, to be compared with the colour segmentation of Figure 1. As we can observe, both texture gradients segment correctly the region of each zebra (which are certainly defined by their texture), such as we wanted. However, we can also note that the contours of the obtained regions are not very precise. Indeed, the segmentation according to a texture gradient gives rough regions.

4. Structural gradient for watershed-based joint colour/texture segmentation

The approach to produce a structural segmentation consists in constructing a joint gradient of colour and texture. Once both a colour gradient and a texture gradient are available, it seems obvious that we can combine them to obtain the called structural gradient. Among the different alternatives for the combination of gradients, we retained two of them which appear particularly simple to implement and sufficiently flexible to evaluate the influence of a gradient with respect to the other. In fact, it deals, on the one hand, with the sum of the colour gradient and a weighted texture gradient (to control the influence of the second one); and on the other hand, with a barycentric linear combination of both gradients. In mathematical terms, we have:

$$\begin{aligned} \varrho_{str}^{I-\alpha}(\mathbf{f})(x) &= \varrho_{col}(\widehat{\mathbf{f}})(x) + \alpha \varrho_{tex}(f_{tex})(x), \\ \varrho_{str}^{II-\alpha}(\mathbf{f})(x) &= (1-\alpha)\varrho_{col}(\widehat{\mathbf{f}})(x) + \alpha \varrho_{tex}(f_{tex})(x), \end{aligned}$$

where $0 \leq \alpha \leq 1$. For both cases, $\varrho_{col}(x)$ and $\varrho_{tex}(x)$ correspond to the definitions previously introduced in the paper. It is obvious that both structural gradients are essentially equivalent; moreover, permitting $\alpha > 1$, identical linear combinations could be obtained. However, as remarked above, the aim of $\varrho_{str}^{I-\alpha}(\mathbf{f})(x)$ is to incorporate the information of texture gradient as a secondary term with respect to the colour, whereas the barycentric formulation $\varrho_{str}^{II-\alpha}(\mathbf{f})(x)$ defines a trade-off between texture/colour gradients. In addition, the inherent normalisation of equation $\varrho_{str}^{II-\alpha}(\mathbf{f})(x)$ preserves the dynamic range of the final gradient image, which could be necessary for watershed computation. We could also consider that the weighting values are not constant for all the image points; or in other words, to define for instance $\varrho_{str}^{II-\alpha}(\mathbf{f})(x) = (1-\alpha(x))\varrho_{col}(\widehat{\mathbf{f}})(x) + \alpha(x)\varrho_{tex}(f_{tex})(x)$, where $\alpha(x)$ is the local weighting function. The appropriate computation of $\alpha(x)$ is out of the scope of this paper.

Figure 4 shows a comparison of segmentation by watershed on the image of zebras according to various structural gradients. We observe that for the texture analysis based on openings/closings as well as for that based on levelings, the balanced structural gradient, i.e., $\varrho_{str-\Gamma\Phi}^{I-\alpha=1}(\mathbf{f})$ and $\varrho_{str-\Lambda}^{I-\alpha=1}(\mathbf{f})$ respectively, improves the segmentations compared to the colour gradient

$\varrho_{col}(\widehat{\mathbf{f}})$. In addition, for this example, we observe also that the best result corresponds to $\varrho^{II-\alpha=0.8}_{str-\Gamma\Phi}(\mathbf{f})$; (texture here is more appropriate than the colour). Moreover, the fact of combining the texture gradient with the colour gradient leads to more precise contours.

To complement the results of our study, we tested the structural gradients on a series of natural colour images and we evaluated the colour segmentation vs. the structural segmentation by watershed. Figure 5 shows four representative images: Examples 1–3 correspond to the segmentation by marker-based watershed (a marker for the object of interest and a marker for the background) and Examples 4–6 correspond to the volume-based segmentation into 50 regions. For each image the segmentation according to the colour gradient and the structural segmentation according to a balanced colour+texture for the two families of texture descriptors that we studied in the paper ($\varrho^{I-\alpha=1}_{str-\Gamma\Phi}$ and $\varrho^{I-\alpha=1}_{str-\Lambda}$) are given. Due to the fact that it is difficult to know *a priori* for an image if it is the colour or the texture which constitutes the most relevant information for the segmentation, we think that the most judicious choice is a balanced combination of this type.

We note that for the example of the butterfly, the structural segmentation is always more coherent than that of the colour. With $\varrho^{I-\alpha=1}_{str-\Gamma\Phi}$, only a part of the wings is obtained (which have same colour-texture) and with $\varrho^{I-\alpha=1}_{str-\Lambda}$ the two colour-textures of the wings are taken into account, producing a perfect segmentation. A similar analysis can be made for the segmentation of the image 2 (a marker following the people and another for the background). In this case, the "texture" corresponds to the head and clothes details of the people. The image of the tiger is a good counterexample which shows that if the texture between the object of interest and the background is very similar, the fact of using a structural gradient will probably introduce a biased segmentation (an intermediate result is obtained using the barycentric linear combination with a greater weight for the colour gradient than for the texture gradient).

For the segmentation of complex images into 50 regions which contain well contrasted coloured objects as well as large areas with or without texture. We see that the structural gradient makes it possible to improve the well-known problem of the volume-based watershed which over-segments the large and homogeneous areas (e.g., sky in Image 4). In addition, certain objects of small size are better segmented with the structural gradients. The contribution of texture allows finding the contours of certain texture zones which are not determined by the colour, as can be observed on Images 5 and 6.

Finally, it is difficult to affirm generally, and without more exhaustive and systematic tests in a large database, if the partitions for $\varrho^{I-\alpha=1}_{str-\Gamma\Phi}$ are more relevant than those for $\varrho^{I-\alpha=1}_{str-\Lambda}$. We observe from the examples that in random textures (e.g., tiger or bear fur, natural textures of Image 6) the segmentation associated to an openning/closing granulometry, $\varrho^{I-\alpha=1}_{str-\Gamma\Phi}$, is more satisfactory. In other examples, where the notion of texture is more related to certain significant structures (e.g., the image of the butterfly), the gradient $\varrho^{I-\alpha=1}_{str-\Lambda}$ from the pseudo-granulometry of connected filters seems to be more appropriate.

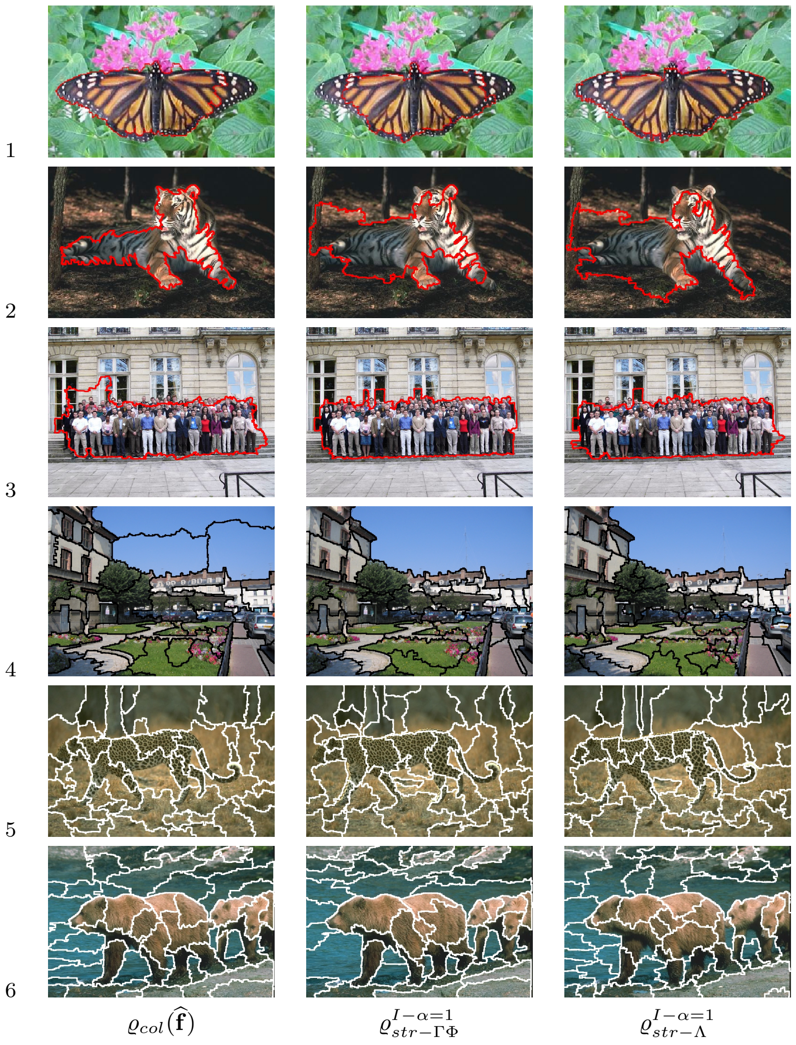

1
2
3
4
5
6
$\varrho_{col}(\widehat{\mathbf{f}})$
$\varrho^{I-\alpha=1}_{str-\Gamma\Phi}$
$\varrho^{I-\alpha=1}_{str-\Lambda}$

Figure 5. Watershed-based colour segmentation versus structural segmentation. Examples 1–3, markers for objects; Examples 4–6, selection of 50 volumic regions.

5. Conclusions and perspectives

This paper presented a morphological approach to calculate texture gradients and it showed how to use them for image segmentation according to the texture; and more generally, for joint colour/texture segmentation (i.e., structural segmentation). We illustrated that these gradients are directly usable for morphological segmentation by watershed and that the partitions obtained with structural gradients are, in most of cases, more relevant than those obtained only with colour gradients. In particular, we showed that the areas of texture are better determined and that the over-segmentation of large and homogeneous zones is reduced.

At present, we are interested in the definition of colour-texture decompositions, without limiting the texture layer to the luminance information. Our purpose is to evaluate the interest of residues of colour openings/levelings (for instance, colour operators defined by means of total orderings in the luminance/ saturation/ hue representation). In addition, we are working on an automatic and local combination of the gradients of colour and texture such that this coupling of information should be adapted to the local image characteristics (i.e., computation of function $\alpha(x)$).

References

[1] J. Angulo, *Morphological colour image simplification by saturation-controlled regional levellings*, Pattern Recognition **20** (2006), no. 8, 1207–1223.

[2] J. Angulo and J. Serra, *Modelling and Segmentation of Colour Images in Polar Representations*, Image Vision and Computing **25** (2007), no. 4, 475–495.

[3] J.-F. Aujol, G. Gilboa, T. Chan, and S. Osher, *Structure-Texture Image Decomposition - Modeling, Algorithms, and Parameter Selection*, International Journal of Computer Vision **67** (2006), no. 1, 111–136.

[4] S. Beucher, *Watershed, hierarchical segmentation and waterfall algorithm*, ISMM'94 (1994), Mathematical Morphology and its Applications to Image and Signal Processing, pp. 69–76.

[5] E. R. Dougherty, J. T. Newell, and J. B. Peltz, *Morphological texture-based maximum likelihood pixel classification based on local granulometric moments*, Pattern Recognition **25** (1992), 1181–1198.

[6] P. R. Hill, C. N. Canagarajah, and D. R. Bull, *Image segmentation using a texture gradient based watershed transform*, IEEE Transactions on Image Processing **12** (2003), no. 12, 1618–1633.

[7] M. A. Hoang, J.-M. Geusebroek, and A. W. M. Smeulders, *Color texture measurement and segmentation*, Signal Processing **85** (2005), 265–275.

[8] W. Y. Ma and B. S. Manjunath, *EdgeFlow: A technique for boundary detection and segmentation*, IEEE Transactions on Image Processing **9** (2000), no. 8, 1375–1388.

[9] P. Maragos, *Pattern Spectrum and Multiscale Shape Representation*, IEEE Trans. on Pattern Analysis and Machine Intelligence **11** (1989), 701–716.

[10] D. R. Martin, C. C. Fowlkes, and J. Malik, *Learning to Detect Natural Image Boundaries Using Local Brightness, Color, and Texture Cues.*, IEEE Trans. on Pattern Analysis and Machine Intelligence **26** (2004), 1–20.

[11] G. Matheron, *Ensembles aléatoires et géométrie intégrale. Tome II*, Ecole des Mines de Paris, Les Cahiers du Centre de Morphologie Mathématique, Fascicule 6, 1972.

[12] F. Meyer, *An Overview of Morphological Segmentation*, International Journal of Pattern Recognition and Artificial Intelligence **15** (2001), no. 7, 1089–1118.

[13] ______, *Levelings, Image Simplification Filters for Segmentation*, Journal of Mathematical Imaging and Vision **20** (2004), 59–72.

[14] F. Meyer and S. Beucher, *Morphological segmentation*, Journal of Visual Communication and Image Representation **1** (1990), no. 1, 21–45.

[15] Y. Meyer, *Oscillating Patterns in Image Processing and Nonlinear Evolution Equations*, AMS, University Lecture Series Vol. 22, 2002.

[16] K. A. Patwardhan and G. Sapiro, *Automatic Image Decomposition*, ICIP'04 (2004), IEEE International Conference on Image Processing, Vol. I, pp. 645–648.

[17] J. Serra, *Image analysis and mathematical morphology, Vol. I*, Academic, London, 1982.

[18] A. Sofou, G. Evangelopoulos, and P. Maragos, *Coupled geometric and texture PDE-based segmentation*, ICIP'05 (2005), IEEE International Conference on Image Processing, Vol. II, pp. 650–653.

[19] C. Vachier, *Morphological Scale-Space Analysis and Feature Extraction*, ICIP'01 (2001), IEEE International Conference on Image Processing, Vol. III, pp. 676–679.

[20] C. Vachier and F. Meyer, *Extinction value: a new measurement of persistence*, (1995), IEEE Workshop on Nonlinear Signal and Image Processing, pp. 254–257.

[21] I. Vanhamel, A. Katartzis, and H. Sahli, *Hierarchical segmentation via a diffusion scheme in color/texture feature space*, ICIP'03 (2003), IEEE International Conference on Image Processing, Vol. I, pp. 969–972.

[22] L. Vese and S. Osher, *Modeling textures with total variation minimization and oscillating patterns in image processing*, J. Sci. Comp. **19** (2003), 553–572.

[23] L. Vincent, *Local grayscale granulometries based on opening trees*, ISMM'96 (1996), Mathematical Morphology and its Applications to Image and Signal Processing, pp. 273–280.

Oversegmentation control for inexact graph matching: First results

Luís A. Consularo[1], Roberto M. Cesar-Jr[2], Luiz H. Figueiredo[3] and Isabelle Bloch[4]

[1] *Universidade Metodista de Piracicaba (UNIMEP), SP, Brazil*
`laconsul@unimep.br`

[2] *Departmento de Ciência da Computação, Instituto de Matemática e Estatística (IME), Universidade de São Paulo, SP, Brazil*
`cesar@ime.usp.br`

[3] *Instituto de Matemática Pura e Aplicada (IMPA), Rio de Janeiro, RJ, Brazil*
`lhf@impa.br`

[4] *Ecole Nationale Supérieure des Télécommunications (GET - Télécom Paris), CNRS UMR 5141 LTCI, Paris, France*
`Isabelle.Bloch@enst.fr`

Abstract An interactive image segmentation method based on structural pattern recognition has been recently introduced. A model graph is generated from an oversegmentation of the image and from traces provided by the user. An input graph is generated from the oversegmented image. Image segmentation is then obtained by matching the input graph to the model graph. An important problem that should be addressed is how to control the size of the input graph. This size is given by the number of regions provided by the oversegmentation. To address this problem, we propose to control the maximum number of regions provided by the oversegmentation by using watershed with markers. The markers are given automatically by using two approaches: quadtrees and centroidal Voronoi diagrams. Results on real images are discussed.

Keywords: inexact graph matching, oversegmentation control, image segmentation quadtrees centroidal Voronoi diagrams.

1. Introduction

Image segmentation is a key problem in most situations in image processing, analysis and computer vision. From the mathematical morphology point-of-view, there are two main paradigms: the flat zone approach using connected filters [4, 8] and the watershed-based methods [15]. Two key problems that often arise in the context of the watershed are regularization and oversegmentation. Methods that address the regularization problem include the viscous watershed transform [14] and watersnakes [10]. On the other hand, oversegmentation may be addressed by watershed with markers [13] (the most popular approach) or by a hierarchical approach [9].

Inexact graph matching represents a structural alternative that has been used for image segmentation [2,3,12]. Since it is a model-based approach, it solves simultaneously the image segmentation and parts recognition problems. In the case of image segmentation, two attributed relation graphs (ARGs) are required. A *model graph* G_m should be available. There are different ways to obtain such models, depending on the application. On the other hand, an *input graph* G_i is generated from an oversegmented image, e.g., by using watershed. Image segmentation is then carried out by matching the input graph to the model graph. The possible graph matches may be shown to be equivalent to cliques of the association graph between input and model graphs. There are $|V_m|^{|V_i|}$ possible solutions for this problem, $|V_m|$ and $|V_i|$ denote the number of vertices of G_m and G_i, respectively. Because of the large number of possible matches, an objective function to assess the quality of each clique must be defined and optimized in order to provide the most suitable image segmentation.

The size of the model graph is controlled by the operator that creates the model. On the other hand, a key problem is how to control the size of the input graph, i.e., $|V_i|$, which is normally given by the number of regions of the oversegmented image. As mentioned above, watershed oversegmentation may be controlled either by markers or by a hierarchical approach. This paper adopts watershed with markers, which is the key difference from the previous methods [2,3,12]. Because the main goal of using the markers in this case is not to identify the desirable objects in the image, but only to control the number of segmented regions, the markers are generated automatically from the image, the only parameter being the maximum number of image regions in the oversegmented image. The other alternative, i.e., hierarchical watershed, will be explored and compared in future research.

We have recently introduced a semi-automated approach for model initialization to guide the graph matching segmentation procedure [3]. The input image to be segmented is decomposed into regions using watershed, as shown in Figure **??**. Some regions of the oversegmented image are manually labeled by traces drawn on the main structures to be segmented. The model graph is automatically derived from the image and the watershed regions intersected by the label traces provided by the user.

This paper is organized as follows. Section 2 reviews our method. The main novelty of the present paper, detailed in Section 3, consists of the automatic generation of markers for watershed using quadtrees and the centroidal Voronoi diagrams. Experimental results are described in Section 4. This paper is concluded with some comments on our ongoing work in Section 5.

2. Model-based image segmentation

We follow the notation described in [3], where more detailed information may be found. $G = (V, E)$ denotes a directed graph where V represents the

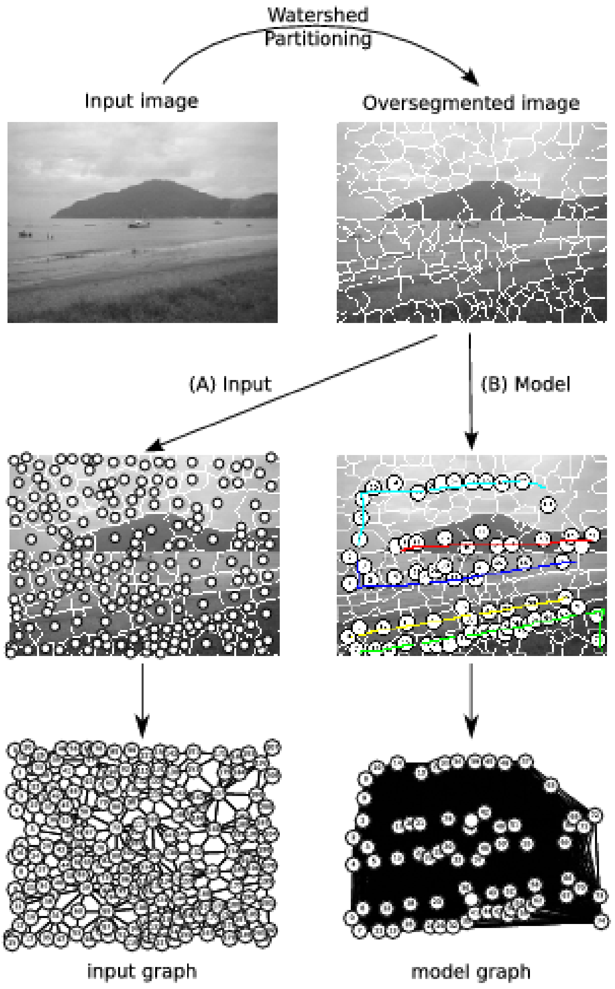


Figure 1. The input and model graphs formation process: the input image is oversegmented by a watershed procedure. Each region is represented as an input graph vertex. An adjacency graph is then generated. The user defines the model graph vertices by drawing label traces on some structurally important regions. The model graph is created as a complete graph. (Adapted from [3]).

set of vertices of G and $E \subseteq V \times V$ the set of edges. An *attributed relational graph* (ARG) is defined as $G = (V, E, \mu, \nu)$, where $\mu : V \to L_V$ assigns an attribute vector to each vertex of V. Similarly, $\nu : E \to L_E$ assigns an attribute vector to each edge of E. The vertices and the edges attributes are called *object* and *relational* attributes, respectively. Two ARGs $G_i = (V_i, E_i, \mu_i, \nu_i)$ and $G_m = (V_m, E_m, \mu_m, \nu_m)$ are adopted, i.e., the *input* (i.e., derived from the image) and the *model* graphs, respectively. $|V_i|$ denotes the number of vertices in V_i, while $|E_i|$ denotes the number of edges in E_i. We use a subscript to denote the corresponding graph, e.g., $a_i \in V_i$ denotes a vertex of G_i, while $(a_i, b_i) \in E_i$ denotes an edge of G_i. We define in this paper $\mu(a) = (g(a))$, where $g(a)$ denotes the average gray-level of the image region associated to vertex $a \in V$, normalized between 0 and 1 with respect to the minimum and maximum possible gray-levels. Let $a, b \in V$ be two vertices of G, and p_a and p_b be the centroids of the respective corresponding image regions. The relational attribute $\nu(a, b)$ of $(a, b) \in E$ is defined as the vector $\nu(a, b) = (p_b - p_a)/(2d_{max})$, where d_{max} is the largest distance between any two points of the input image region.

An inexact match between G_i and G_m may be represented as an approximate homomorphism between G_i and G_m and is searched in the corresponding association graph [2]. The *association graph* G_A between G_i and G_m is defined as the complete graph $G_A = (V_A, E_A)$, with $V_A = V_i \times V_m$. An inexact match between G_i and G_m can be expressed as a clique (i.e., a complete subgraph) $G_S = (V_S, E_S)$ of the association graph G_A between G_i and G_m with $V_S = \{a_{im} = (a_i, a_m), a_i \in V_i, a_m \in V_m\}$ such that $\forall a_i \in V_i, \exists a_m \in V_m, a_{im} \in V_S$ and $\forall a_{im} \in V_S, \forall b_{im} \in V_S,\ a_i = b_i \Rightarrow a_m = b_m$ which guarantees that each vertex of the image graph has exactly one label (i.e., it is mapped onto a single vertex of the model graph) and $|V_S| = |V_i|$.

There is a huge number of cliques that represent possible inexact matches between G_i and G_m, namely $|V_m|^{|V_i|}$. The evaluation of the quality of a solution expressed by G_S is performed through an objective function:

$$f(G_S) = \frac{\alpha}{|V_S|} \sum_{a_{im} \in V_S} c_V(a_{im}) + \frac{(1-\alpha)}{|E_S|} \sum_{e \in E_S} c_E(e), \tag{1}$$

where $c_V(a_{im})$ is a measure of dissimilarity between the attributes of a_i and a_m. Similarly, if $e = (a_{im}, b_{im})$, $c_E(e)$ is a measure of the dissimilarity between edge (a_i, b_i) of the image and edge (a_m, b_m) of the model. The dissimilarity objective function should therefore be minimized. Let $a_{im} \in V_A$, $a_i \in V_i$ and $a_m \in V_m$. The dissimilarity measure $c_V(a_{im})$ is defined as $c_V(a_{im}) = |g_i(a_i) - g_m(a_m)|$, where $g_i(a_i)$, $g_m(a_m)$ are the object attributes of vertices $a_i \in G_i$, $a_m \in G_m$, respectively. Let $e = (a_{im}, b_{im}) \in E_A$. We compute the modulus and angular differences between $\nu(a_i, b_i)$ and $\nu(a_m, b_m)$ as $\varphi_m(e) = |\|\nu(a_i, b_i)\| - \|\nu(a_m, b_m)\||$ and $\varphi_a(e) = \frac{|\cos(\theta)-1|}{2}$, respectively, where θ is the angle between $\nu(a_i,\, b_i)$ and $\nu(a_m, b_m)$, i.e., $\cos(\theta)$ is calculated as:

$$\cos(\theta) = \frac{\nu(a_i, b_i) \cdot \nu(a_m, b_m)}{\|\nu(a_i, b_i)\| \|\nu(a_m, b_m)\|}.$$

In order to define the dissimilarity measure $c_E(e)$, we need an auxiliary function: $\hat{c}_E(e) = \gamma_E \varphi_a(e) + (1 - \gamma_E) \varphi_m(e)$. The parameter γ_E $(0 \leq \gamma_E \leq 1)$ controls the weights of φ_m and φ_a. It is important to note that $\nu(a, a) = \vec{0}$.

This fact means that, when two vertices in G_i are mapped onto a single vertex of G_m, we have $c_E(e) = \|\nu(a_{i_1}, a_{i_2}) - \vec{0}\| = \|\nu(a_{i_1}, a_{i_1})\|$, which is proportional to the distance between the centroids of the corresponding regions in the oversegmented image (in such cases, we define $\cos(\theta) = 1$). Therefore, $\hat{c}_E$ provides large dissimilarity values when assigning the same label (i.e., the target vertex in G_m) to distant regions and lower values when assigning the same label to near regions, which is intuitively desirable in the present application.

Let $a_{i_1}, a_{i_2} \in V_i$ and $a_{m_1}, a_{m_2} \in V_m$ be vertices of G_i and G_m, respectively. Suppose that a_{i_1} and a_{i_2} are matched to a_{m_1} and a_{m_2}, respectively. In this case, the edge (a_{i_1}, a_{i_2}) should be matched to (a_{m_1}, a_{m_2}) and the dissimilarity measure between them should be evaluated. However, depending on the adopted graph topology, it is possible that one or both edges do not actually exist and the dissimilarity measure should properly deal with such situations. The edge dissimilarity measure is therefore defined by Equation 2. It is important to highlight the case of $e' = (a_{i_1,m_1}, a_{i_2,m_2})$. The edge comparisons depend on the graph topology adopted for G_m and G_i. In the present case, an adjacency graph has been adopted for G_i whereas a complete graph is generated for G_m. Therefore, there are matching situations where there exists an edge in G_m but not on G_i (i.e., $(a_{i_1}, a_{i_2}) \notin E_i, (a_{m_1}, a_{m_2}) \in E_m$) because of the different adopted graph topologies. In this case, the (a_{i_1}, a_{i_2}) features are calculated on-the-fly using the same procedure to calculate the edge features, thus allowing the comparison to (a_{m_1}, a_{m_2}). This problem could be solved by adopting a complete graph for G_i, as in our previous works, but this would naturally increase the memory costs.

$$c_E(e) = \begin{cases} \hat{c}_E(e), & \text{if } (a_{i_1}, a_{i_2}) \in E_i, (a_{m_1}, a_{m_2}) \in E_m, \\ \hat{c}_E(e'), & \text{if } (a_{i_1}, a_{i_2}) \notin E_i, (a_{m_1}, a_{m_2}) \in E_m, \\ \infty, & \text{if } (a_{i_1}, a_{i_2}) \in E_i, (a_{m_1}, a_{m_2}) \notin E_m, \\ 0, & \text{if } (a_{i_1}, a_{i_2}) \notin E_i, (a_{m_1}, a_{m_2}) \notin E_m. \end{cases} \tag{2}$$

The objective function (Equation 1) should be optimized in order to find a suitable inexact match between G_i and G_m. There are many different optimization algorithms that may be used and the reader is referred to [2] for a comparative review that includes beam-search, genetic algorithms and Bayesian networks. The results presented in this paper have been obtained using the clique-search algorithm described in [3]. The optimization algorithm starts with an empty clique G_S and incrementally increases it by evaluating the objective function (Equation 1). The cheapest clique is chosen and a new vertex is added to it at each iteration. The algorithm stops when a clique that represents a complete solution is found. The vertices of G_A are of the form $a_{im} = (a_i, a_m), a_i \in V_i, a_m \in V_m$. For each $a_i \in V_i$ there is a set of vertices $a_{im} = (a_i, a_m), a_m \in V_m$ that represents all possible labels to which a_i may be assigned. Each of these sets is called a *supervertex* of G_A, defined as:

$$s_i = \{a_{im} = (a_i, a_m) \in V_s, a_i \in V_i, \forall a_m \in V_m\}.$$

A clique G_S that represents a valid solution is composed by one single vertex a_{im} of each supervertex s_i in G_A. For each supervertex, the association vertex a_{im} with the best node cost defines the supervertex cost. The proposed algorithm selects the cheapest supervertex s_i at each iteration. All vertices a_{im} of the selected supervertex s_i are considered in order to identify the one minimizing the objective function (Equation 1) when added to the solution clique. This idea is inspired by the Sequential Forward Search (SFS) algorithm for feature selection [7]. The final solution produced by the matching procedure may be represented as a labeled image where a label associated to the model vertices is assigned to each pixel (actually, to all pixels of each watershed connected region). A mode filter is applied to the labeled image to smooth the produced boundaries and to eliminate small noisy labels.

3. Markers detection

In order to find markers suitable for the watershed, we use quadtrees and centroidal Voronoi diagrams, an approach originally proposed for generating mosaic effects [6]. The simplest method for finding markers is by sampling the image adaptively using a quadtree. The stop criterion in the quadtree is that the intensity of all pixels in a cell is close to the average intensity in the cell. To avoid getting to single-pixel cells, we also stop the subdivision when cells get too small. The markers are the centers of the leaf cells in the quadtree. This quadtree sampling method generates two kinds of points: points that are clustered around the image edges, and points in the middle of homogeneous regions (having a low level of detail). Both kinds of points are needed for a fair sampling of the image. The important aspect is that the center of different cells tends to be on different image basins, thus defining potentially good markers for the watershed. Because only a maximum number of quadtree cells are available, oversegmentation may be controlled by the allowed total number of cells.

A more sophisticated method for finding markers is to use a centroidal Voronoi diagram whose sites are adjusted to the image features. The *Voronoi diagram* of a set of points, called *sites*, is a decomposition of the space into cells, one cell for each site, such that the cell corresponding to a site p is the set of all points in space which are closer to p than to any other site [1, 11]. A *centroidal Voronoi diagram* is a Voronoi diagram in which each site is the centroid of its cell [5].

Centroidal Voronoi diagrams are very rare. However, any Voronoi diagram can be transformed into a centroidal Voronoi diagram by a simple relaxation procedure known as *Lloyd's algorithm*: replace each Voronoi site by the centroid of its Voronoi cell, recompute the Voronoi diagram for the new sites, and repeat until convergence.

A major ingredient in the definition of a centroidal Voronoi diagram is an underlying *density function* with respect to which the centroids of the Voronoi cells are found. More precisely, the centroid of a region V with respect to a density function μ is the point z given by

$$z = \frac{\displaystyle\int_V x\mu(x)\,dx}{\displaystyle\int_V \mu(x)\,dx} \,.$$

Naturally, for images we use sums over pixels as the discrete analogues of these integrals. The density function μ does not enter in the computation of the Voronoi diagram, which is still computed using the Euclidean metric.

Centroidal Voronoi diagrams adapt themselves to the mass distribution implied by the density function, having larger cells where the density is low and smaller cells where the density is high. As in the mosaic approach described in [6], we use the Euclidean norm of the gradient of the image as density function. The gradient is computed using central differences. We start from the markers found in the quadtree sampling step and compute a centroidal Voronoi diagram using a few iterations of Lloyd's algorithm. The new markers are the centroids of the final Voronoi diagram.

4. Experimental results

The proposed approach has been applied to different images of the Berkeley Image Segmentation Database[1]. The traced strokes manually defined for three test images are shown in Figure 2. Each color represents a different label to be recognized by the matching procedure. In order to compare the different approaches, the same set of traced strokes for each test images have been used to generate the models for the three approaches assessed in the present paper: (i) watershed without markers; (ii) watershed with markers generated by quadtrees; (iii) watershed with markers generated by the centroidal Voronoi diagrams. It is worth noting that the image regions may present similar gray-level and belong to different model classes defined by the user labels. Also, there are some image regions with substantial gray-level variation because of belonging to non-homogeneous textured regions, which are traditionally very difficult to segment. The structural information leads to a robust segmentation performance even in such cases.

The segmentation results are shown in Figures 3, 4 and 5. These figures present, for each image, the results using the watershed without markers

[1] http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

(top row), the watershed with markers provided by the quadtree (middle row) and the watershed with markers provided by the centroidal Voronoi diagram (bottom row). The first column shows the oversegmentation in the case of the watershed without markers and the corresponding quadtree and Voronoi partitions with the cells seeds, which are used as markers for the watershed. The middle column shows the corresponding watershed partitions (red) and the final segmentation result in green. Finally, the last row shows the final segmentation results with the region labels provided by the inexact matching procedure.

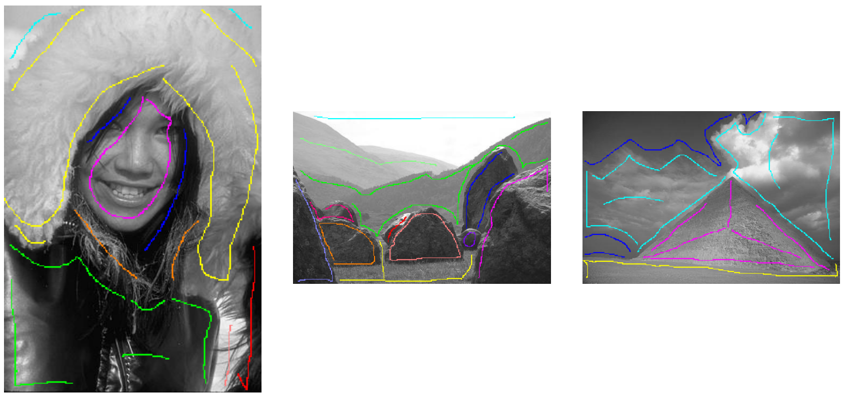


Figure 2. Strokes traced by the user on each test image are used to generate the graph models. Each color is associated to a different label, i.e., a different class to be recognized by the matching procedure.

As it can be seen, though a smaller number of larger regions are used by the matching procedure in the case of the watershed with markers (both quadtree and Voronoi), the final segmentation results are comparable. In some cases, the segmentation actually improves, once the watershed without markers lead to too many regions, some very small, which makes more difficult the correct matching and may lead to misclassifications. Therefore, in general, the final segmentation is comparable for any of the three assessed approaches. Nevertheless, the main difference lies in the memory and running time costs, as shown in Table 1. It can be seen from that table that the input and model graphs are substantially smaller than with the previous approach without markers. The running time differences decrease from nearly an hour to some seconds. This important decrease in running time is explained because the total number of possible solutions is exponential in the sizes of the graphs ($|V_m|^{|V_i|}$) and, once both $|V_i|$ and $|V_m|$ decrease, the search space is considerably shrinked. It is also important to note that, although the centroidal Voronoi method performs a kind of fine tuning in the position of the markers, there is apparently no strong differences between the final result produced by it and the results produced by the quadtrees.

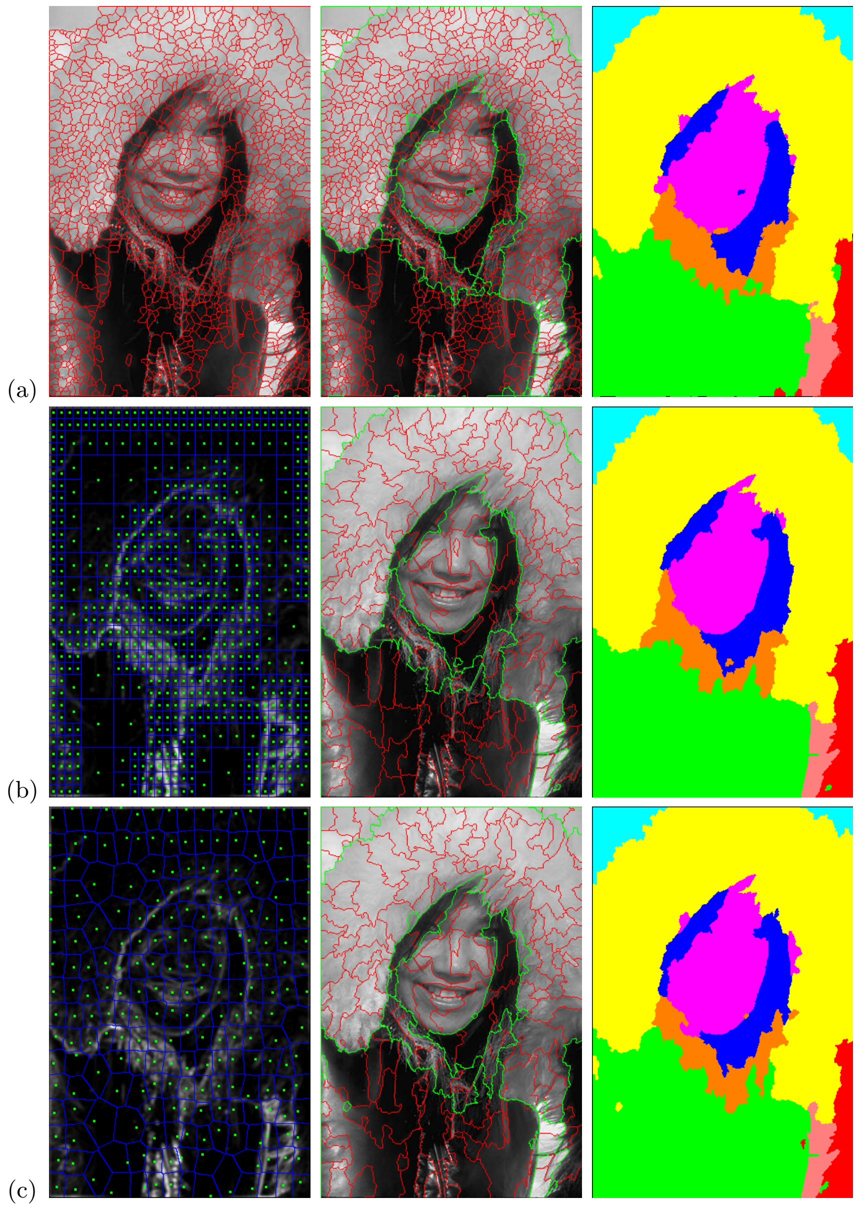


Figure 3. (a) Graph matching without markers; (b) Graph matching with quadtree markers; (c) Graph matching with Voronoi markers.

Once the quadtrees approach is computationally cheaper, it may be more

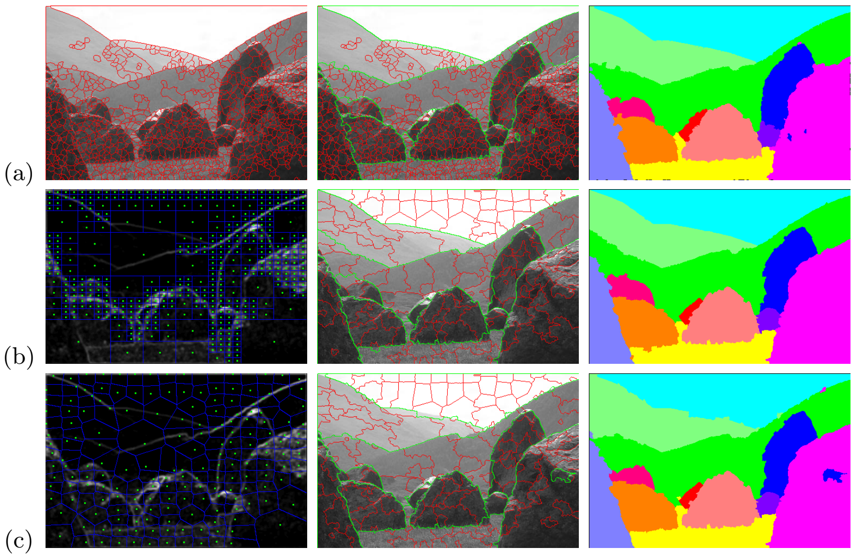


Figure 4. (a) Graph matching without markers; (b) Graph matching with quadtree markers; (c) Graph matching with Voronoi markers.

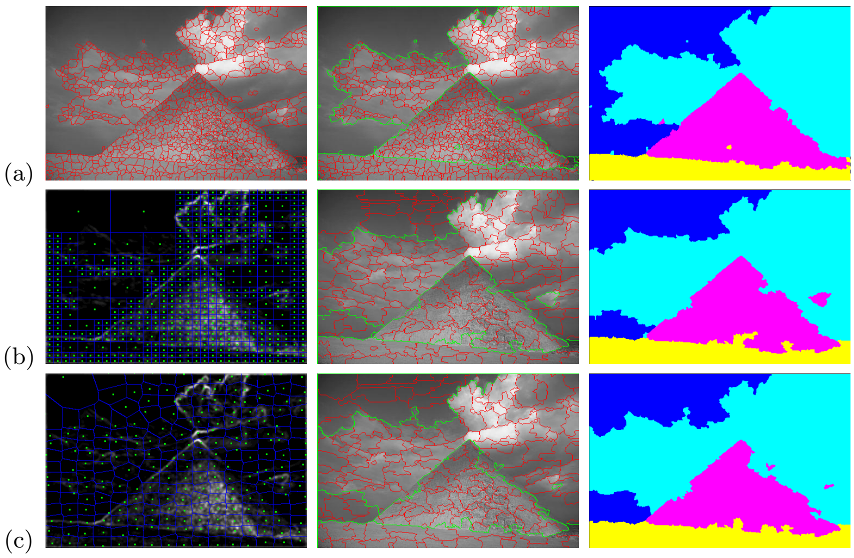


Figure 5. (a) Graph matching without markers; (b) Graph matching with quadtree markers; (c) Graph matching with Voronoi markers.

advantageous. However, because different density functions may be adopted for the centroidal Voronoi diagram, we feel that a good research topic is to look for more suitable functions that could improve its performance. For instance, an obvious drawback of the quadtree/Voronoi approaches is that, because the oversegmented image is partitioned in a smaller number of regions, some true edges are lost. It would be nice to have a method to avoid such a problem.

Table 1. The table summarizes the size of the input and model graphs ($|V_i|$ and $|V_m|$, respectively) and the running time to segment the image.

Watershed	Figure	$\|V_i\|$	$\|V_m\|$	Running time
Without markers	3(a)	1482	352	1h34m20s
Quadtree	3(b)	226	152	25s
Voronoi	3(c)	226	133	24s
Without markers	4(a)	1102	373	59m30s
Quadtree	4(b)	187	121	14s
Voronoi	4(c)	187	117	13s
Without markers	5(a)	1003	317	35m10s
Quadtree	5(b)	232	152	30s
Voronoi	5(c)	232	134	27s

5. Conclusion

We have recently introduced an interactive image segmentation approach based on inexact graph matching. The present paper improves our previous works by exploring the watershed with markers automatically generated by using quadtrees and centroidal Voronoi diagrams. The results are equivalent in quality, but the improvement expressively decreased running time and memory requirements. Our ongoing work includes research to find more suitable density functions for the centroidal Voronoi diagram and the inclusion of additional object attributes such as color and texture. Also, we intend to explore the method for the recognition of object parts by using different model and input images.

References

[1] F. Aurenhammer, *Voronoi diagrams—a survey of a fundamental geometric data structure*, ACM Computing Surveys **23** (1991), no. 3, 345–405.

[2] R. M. Cesar-Jr., E. Bengoetxea, I. Bloch, and P. Larrañaga, *Inexact graph matching for model-based recognition: Evaluation and comparison of optimization algorithms*, Pattern Recognition **38** (2005), no. 11, 2099–2113.

[3] L. A. Consularo, R. M. Cesar-Jr., and I. Bloch, *Structural image segmentation with interactive model generation*, Proc. IEEE International Conference on Image Processing (ICIP-07), 2007.

[4] J. Crespo, R. W. Schafer, J. Serra, C. Gratin, and F. Meyer, *The flat zone approach: a general low-level region merging segmentation method*, Signal Process. **62** (1997), no. 1, 37–60.

[5] Q. Du, V. Faber, and M. Gunzburger, *Centroidal Voronoi tessellations: Applications and algorithms*, SIAM Review **41** (1999), no. 4, 637–676.

[6] G. M. Faustino and L. H. de Figueiredo, *Simple adaptive mosaic effects*, Proceedings of SIBGRAPI'05, 2005, pp. 315–322.

[7] A. Jain and D. Zongker, *Feature selection - evaluation, application, and small sample performance*, IEEE Transactions on Pattern Analysis and Machine Intelligence **19** (1997), no. 2, 153–158.

[8] A. Meijster and M. H. F. Wilkinson, *A comparison of algorithms for connected set openings and closings*, IEEE Trans. Pattern Anal. Mach. Intell. **24** (2002), no. 4, 484–494.

[9] L. Najman and M. Schmitt, *Geodesic saliency of watershed contours and hierarchical segmentation*, IEEE Trans. Pattern Anal. Mach. Intell. **18** (1996), no. 12, 1163–1173.

[10] H. T. Nguyen, M. Worring, and R. van den Boomgaard, *Watersnakes: Energy-driven watershed segmentation.*, IEEE Trans. Pattern Anal. Mach. Intell. **25** (2003), no. 3, 330–342.

[11] A. Okabe, B. Boots, and K. Sugihara, *Spatial tessellations: Concepts and applications of Voronoi diagrams*, John Wiley & Sons, Inc., 1992.

[12] A. Perchant and I. Bloch, *A New Definition for Fuzzy Attributed Graph Homomorphism with Application to Structural Shape Recognition in Brain Imaging*, Imtc'99, 16th ieee instrumentation and measurement technology conference, 1999, pp. 1801–1806.

[13] P. Soille, *Morphological image analysis: Principles and applications*, Springer Verlag, 1999.

[14] C. Vachier and F. Meyer, *The viscous watershed transform*, J. Math. Imaging Vis. **22** (2005), no. 2-3, 251–267.

[15] L. Vincent and P. Soille, *Watersheds in digital spaces: An efficient algorithm based on immersion simulations*, IEEE Trans. Pattern Anal. Mach. Intell. **13** (1991), no. 6, 583–598.

Segmentation of random textures by morphological and linear operators

Aurélien Cord, Dominique Jeulin and Francis Bach

Centre de Morphologie Mathématique (CMM), Fontainebleau, France
`{aurelien.cord,Dominique.Jeulin}@cmm.ensmp.fr, francis.bach@mines.org`

Abstract We propose a linear and a morphological approach for the characterization and segmentation of binary and digital random textures. We focus on descriptors at the level of pixels in images, combined with statistical learning to select and weight them. The approach is illustrated on simulations of texture patchworks, for which errors of classification can be evaluated.

Keywords: texture segmentation, random textures, morphological operations, multivariate statistical analysis, linear discriminant analysis, machine learning.

1. Introduction

Image segmentation is a common and important task aimed at extracting objects that can be subsequently measured or analyzed. This is in particular useful when looking for defects in materials or for pathological cells in a biological tissue. In many cases, the background and the objects themselves are non uniform and made of so-called "textures". A challenging problem concerns the automatic extraction (or segmentation) of various textures present in the same image.

For this purpose, a variety of statistical techniques have been used, such as histogram-based texture analysis techniques corresponding to the use of co-occurrence matrices [6], texture modeling [3], filtering approaches [10] and wavelet transformations of images [9, 11, 14]. However on materials displaying complex patterns that are random in appearance (i.e., not periodic), segmenting texture turns out to be difficult [17].

To handle it, we propose the following approach: to describe the texture for each pixel of the image, accounting for morphological information in its neighborhood at different scales and then using a multivariate statistical approach. This approach is illustrated and validated for binary and gray level textures through simulations.

G. Banon, et al. (eds),
Mathematical Morphology and its Applications to Signal and Image Processing, 387–398.
©2007 *MCT/INPE. Printed in Brazil.*
URL of this article: http://urlib.net/dpi.inpe.br/ismm@80/2007/03.26.17.28

2. Theoretical approach of random textures

Natural textures have the following common characteristics: a texture usually shows fluctuations at a small scale, and some uniformity at a large scale. The presence of fluctuations requires the use of a probabilistic approach to characterize textures. In this framework we will consider binary textures and gray level textures as realizations of random sets or of random functions. Therefore, from a theoretical point of view, a random texture is completely known from its Choquet capacity.

It can be shown [13] that a random closed set A is known from the Choquet capacity functional $T(K)$ defined on compact sets K:

$$T(K) = 1 - P\{K \subset A^c\}, \tag{1}$$

where P is the probability of the event $\{\}$. Similarly, an upper semi-continuous random function (RF) $Z(x)$ is characterized by the functional $T(g)$, defined for test functions g with a compact support K [7,8]:

$$T(g) = P\{x \in D_Z(g)\}, \tag{2}$$

with

$$D_Z(g)^c = \{x\ ,\ Z(x+y) < g(y)\ ,\ \forall\, y \in K\}. \tag{3}$$

When the compact set K is a point x and $g(x) = z$, the cumulative distribution function is obtained. When using the two points $\{x, x+h\}$ and the function defined by $g(x) = z_1$ and $g(x+h) = z_2$, we can derive the bivariate distribution $F(z_1, z_2, h)$. More generally, using $g(x) = z$ for $x \in K$ and $g(x) = +\infty$ for $x \notin K$ we obtain the distribution function of $Z(x)$ after a change of support according to the supremum over any compact set K.

From this theoretical background, it turns out that good candidates for texture descriptors can be provided by estimates of probabilities obtained after dilations (or erosions) by compact sets for binary textures. Similarly, distribution functions after dilations (or erosions) of gray level images will provide texture descriptions in the digital case. Changing the shape and size of the compact set K provides us with a full set of data.

From a practical point of view, estimates of the Choquet capacity $T(K)$ or $T(g)$ can be obtained on images. The pertinence of descriptors will depend on the statistical precision of estimates.

In a second step, a selection of the more efficient descriptors must be made. This is usually performed by multivariate analysis. This approach was successfully applied to the classification of images in the standard case when there is a single texture per field of view [1,5].

To address the problem of segmentation of textures in images, a classification of pixels must be performed. This requires a local characterization, that can be made in different ways:

1. Transform the image by dilations, erosions by K_i, g_i ("pixel" approach), and generate a multispectral image from the collection of K_i or g_i. It corresponds to filters, like granulometries, as already proposed by [1,5,16]. It provides a set of transformed binary or gray level images (Section 3.2).

2. Consider a neighborhood $B(x)$ of each pixel, and use a local estimate of $T(K)$, $T(g)$ inside $B(x)$. From the estimates, generate a multispectral image from the collection of K_i or g_i. It uses a local estimate in B of the Choquet capacity and requires the appropriate choice of B (Section 3.3).

3. Use measures μ_i with a compact support K_i and estimate $\mu_i(A)$ or $\mu_i(Z)$, generating a multispectral image from the collection of K_i. Particular cases are given by various types of linear filters, like multiscale convolution by Gaussian kernels, wavelets, curvelets, but also local measurements of the Minkowski functionals for a random set A (Section 3.1).

3. Pixel texture description

In this work, texture properties are calculated for each pixel taking into account local properties of its neighborhood at varying scales. This provides us with 3 dimensional data having two spatial dimensions and a texture descriptor dimension. This allows us to characterize texture at the pixel level and follows the texton approach of [12]. Two families of descriptors are used: curvelets and morphological transformations.

3.1 The curvelet transform

The curvelet transform is a higher dimensional generalization of the wavelet transform, designed to represent images at different scales and different angles [2]. The use of this tool to characterize the texture in an image is recent [4]. Curvelets have very interesting properties in the context of object detection, in particular curved singularities can be well approximated with very few coefficients. This makes the curvelets coefficients for pixels belonging to a particular object very specific.

The curvelet filter bank is in essence a set of bandpass filters with range and orientation selective properties. Typically, we apply a linear filtering of each 100×100 neighborhood of every pixel by curvelets with different frequencies and orientations. The filter bank is decomposed into 4 sets of frequency containing 1, 8, 16 and 1 filters of varied orientations from the smallest frequency to the largest one. A spatial filtering of each of the 26 filters results is applied to calculate the local energy function. It aims to identify areas where the band pass frequency components are strong, after conversion into gray levels. The outputs of this function are a first class of texture descriptors.

3.2 Morphological transformations

The morphological operators consist of non linear image transformations [15]. Different structuring elements are chosen: disks, vertical and horizontal segments. For a given type of structuring element and a list of sizes, all eroded and dilated images are evaluated. The descriptor is obtained by calculating the difference between the eroded images at the step n and $n+1$ as well as between the dilated images at steps $n+1$ and n. Therefore each pixel is described by a vector with k morphological components, where $k =$ number of sizes $\times$ number of structuring elements $\times$ 2 operations.

More complex pixel descriptors can also be proposed by using opening and closing operations instead of erosions and dilations, respectively, as made in [1, 5, 16].

In this study, we choose the following morphological descriptors:

- small scales, erosion and dilation with series of sizes of $[1, 2, 3, 4, 5, 7, 9, 12] \times 2 + 1$ (size 48);
- small scales, opening and closing with series of sizes of $[1, 2, 3, 4, 5, 7, 9, 12] \times 2 + 1$ (size 48);
- large scales, opening and closing with series of sizes of $[2, 4, 8, 16, 32] \times 2 + 1$ (size 30).

3.3 Averaged descriptors

From the initial pixel descriptors, it is easy to estimate average descriptors in a window B, in order to use the local Choquet capacity. The best size of the window will be obtained by optimizing the pixels classification. In what follows, averages in windows up to 60×60 are tested.

4. Application

The present application is based on simulations of models of random textures. They are produced and mixed together. Using the knowledge of the ground truth to calculate errors of classification, we are able to test different pixel descriptors for the segmentation. We produce a critical analysis of the approach and evaluate its limitations. Images used here are available on `www.cmm.ensmp.fr/~cord/Synthetik_Texture/` for comparison with any other segmentation approach.

4.1 Data

On the basis of [7,8], we simulate different textures in images of size 800×800 pixel (Figure 1), using the Micromorph© software:

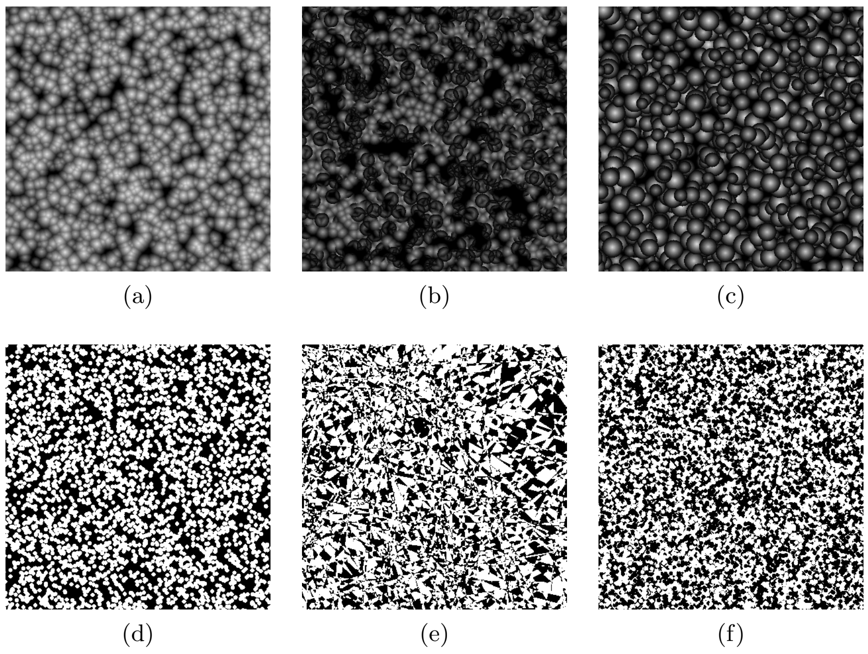


Figure 1. Simulated texture images. (a) Boolean random function. (b) Sequential Alternate random function. (c) Dead leaves random function. (d) Boolean random set, with disk primary grain (radius 6). (e) Poisson mosaic. (f) Binary dead leaves.

- a boolean random function with cone primary function (Figure 1(a)); using a uniform distribution of radii between 1 and 32 pixels.
- a sequential alternate random function with cone primary function (Figure 1(b));
- a dead leaves random function (Figure 1(c));
- a boolean random set, with disk primary grain (radius 6) (Figure 1(d));
- a Poisson mosaic (from a Poisson line tessellation involving 400 lines (Figure 1(e));
- a binary dead leaves with disks (radius 6) (Figure 1(f)).

We consider 4 studied cases, corresponding to 4 patchwork images with a mixture of two textures. They are produced using two original images from Figures 1(a–f). To combine those texture images, we simulate another image of size 800 $\times$ 800 pixel corresponding to a a binary Poisson mosaic with 50 lines presented in Figure 2(e). This image is used as a binary mask. The resulting patchwork images, shown in Figure 2, are used to test the segmentation procedure on the basis of the local textural properties. We can notice that the resulting composite images show some very small areas with different textures, as compared to the size of the primary grains, for which the segmentation is a very difficult task.

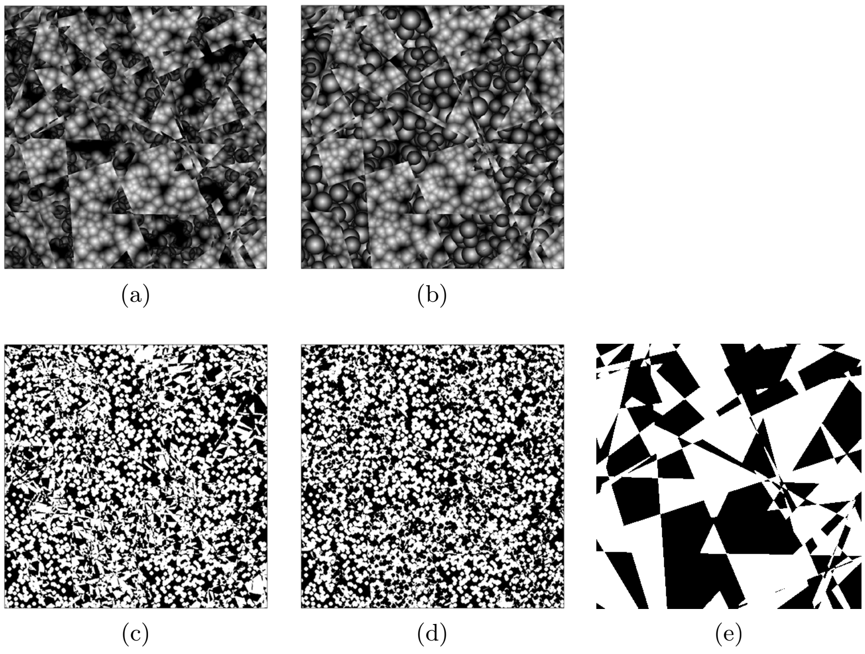


Figure 2. Patchwork image containing two types of texture for separation. The mixtures are obtained by using (e) as a mask. Combination of: (a) Figure 1(a) and 1(b). (b) Figure 1(a) and 1(c). (c) Figure 1(d) and 1(e). (d) Figure 1(d) and 1(f).

4.2 Experiments

Learning Procedure Texture descriptors are calculated both on original images and on the patchwork image. For each patchwork image, two different learning training procedure based on a linear discriminant analysis (LDA) are produced and compared. In each case, both training and testing pixels are extracted. First, we randomly extract from each of the two original images (Figure 1), 20,000 pixels half for the training and the rest for the test. This is labeled "Training ORI" in the following. This analysis aims to calculate the projection of descriptors that corresponds to the best linear separation between the two textures. Second, we extract from half of the patchwork image (Figure 2), 10,000 pixels of each texture for the training and from the other half of the patchwork image 10,000 pixels of each texture for the test. This is labeled "Training PATCHW" in the following. This analysis aims to calculate the projection of descriptors that allows to best discriminate between the two textures even in border areas. We have to stress the fact that the textures in the original images (Figure 1) and in the patchwork images (Figure 2) have different probabilistic properties, since the second ones are a combination of simple textures and of a random set belonging to the mask (the large Poisson mosaic).

An histogram of train data projections on the first LDA axis is presented in Figure 3 for the two learning procedures.

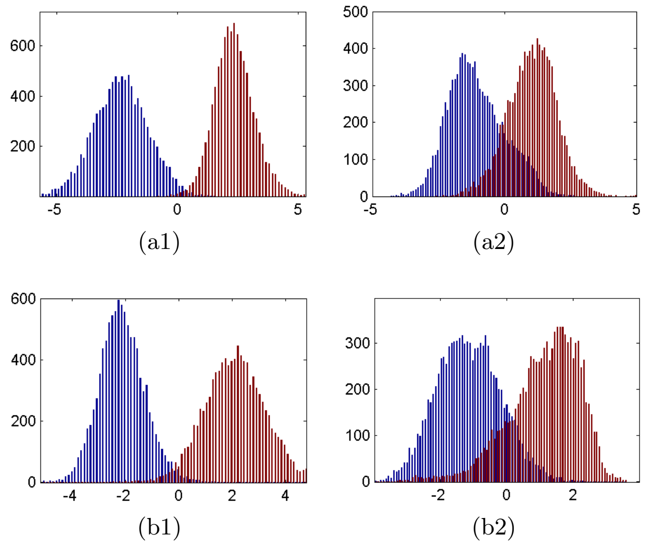


Figure 3. Exemple of histogram of training data projection on the first LDA axis when training on (a1 and b1) ORI, (a2 and b2) PATCHW. (a1 and a2) For gray scale image corresponding to Figure 2(a) (no averaging of the descriptors). (b1 and b2) For binary image corresponding to Figure 2(c) (descriptors are averaged by a box of size 35×35).

It shows that the separation between the two textures is systematically reachable. The separation appears to be more efficient for training ORI than training PATCHW. In this last case, the descriptors for one texture is clearly influenced by the descriptors from the other ones, in particular for pixels located near boundaries. It confirms that textures have different probabilistic properties. The Gaussian shape suggests that LDA was indeed a good method, much simpler to perform than SVM (however we provide a comparison of the results obtained for the two techniques). It leads us to model the data as a mixture of two Gaussian random variables.

Finally, the patchwork image descriptors are projected on the first LDA axis and classified. The probability for each pixel of belonging to one of the Gaussian distribution is calculated and is used to produce the classification. Using the knowledge of the mask, a patchwork error corresponding to a misclassification is evaluated. It is the global error of the approach.

The difference between train and test errors are smaller than 1.5 % in all the cases, showing that the system do not overfit. Therefore we restrict the presentation to test errors.

Different sizes of windows B are used to average descriptors as detailed in Section 3.3. Figure 4 presents the test errors and the patchwork errors as a function of this size for the two learning procedures (training ORI and PATCHW).

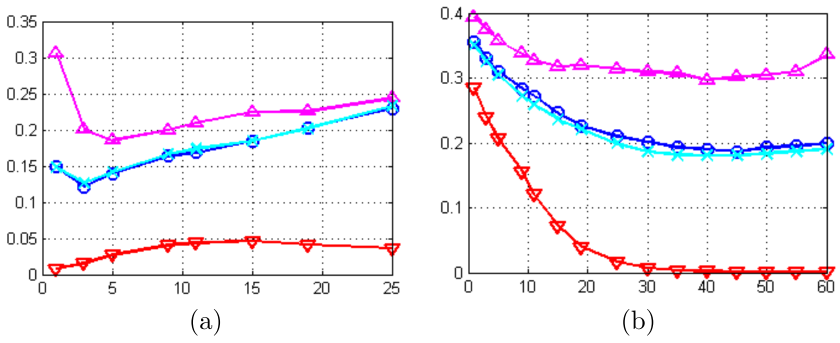


Figure 4. Errors versus size of descriptor averaging boxes (a) For gray scale image corresponding to Figure 2(a). (b) For binary image corresponding to Figure 2(d). Notation: ∇ test error for training ORI. $\triangle$ patchwork error for training ORI. O test error for training PATCHW. X patchwork error for training PATCHW.

Only two of the four projections are presented, because the results are similar for the two gray level images and for the two binary images. However, the behaviors are very different between gray scale and binary images used in this study.

Training on original versus patchwork images

The gap between test errors and patchwork errors for training ORI shows that this training procedure produces a large error, as a result of edge effects induced by the boundaries of domains with a single texture. Training on images with a single texture is therefore not very efficient, and should be avoided for applications to texture segmentation when they are intimately mixed.

Influence of the window size of the post-averaging

(a) For gray scale images, only a small box of size 3×3 slightly improves the results. It shows that the descriptors on the scale of pixels are well adapted to our problem in the case of the gray level images.

(b) For the training on original images, we note that increasing the averaging window size systematically decreases the test error: the separation of the textures increases, as a result of a better estimate of the descriptors. However, the precision of the detection decreases, in particular near the boundaries. Therefore we have to find a trade-off, corresponding to the minimum reached on the patchwork test error curve.

In the following, we keep no averaging for the gray scale and a 35×35 averaging box for the binary images. A summary of the test and patchwork errors is given in Table 1.

Table 1. Test and patchwork errors for the different proposed procedures. Average box sizes 35×35. All values are in %.

Figure	Training ORI				Training PATCHW				
	No Average		Average		No Average		Average		SVM
	Test	Patch	Test	Patch	Test	Patch	Test	Patch	Patch
2(a)	1.0	29.5	NA	NA	15.5	**15.7**	NA	NA	9.7
2(b)	0.8	30.5	NA	NA	14.9	**14.8**	NA	NA	9.4
2(c)	25.6	30.6	1.8	15.8	29.1	29.2	13.2	**13.5**	12.1
2(d)	28.4	39.4	3.7	30.7	35.6	35.0	19.3	**18.1**	19.3

Results analyses

With the present approach we obtain between 82 % and 90 % of pixels that are correctly classified. These results are very satisfying taking into account the complexity of the chosen images. We run a SVM using a radial basis function ($\exp(-\frac{1}{2}(u-v)^2)$) for the best training procedure (train PATCHW with the adapted averaging window size). By minimizing the test error, we optimized the regularization parameter C that controls the trade-off between training set accuracy and generalization performance. The result, presented in Table 1 are similar to LDA ones for the binary images and greatly improved for gray scale images.

Chosen descriptors

We have also looked for descriptors giving a larger contribution to the LDA projection. The disk structuring element appears to be the most relevant one for all present applications, where the textures are isotropic. The erosion/dilation are most relevant for binary images and opening/closing for the gray level images. The present curvelet approach, which acts symmetrically on images and on their negative, is unable to separate textures like the sequential alternate and the Boolean random function. Indeed, the curvelets are more relevant for binary images than grey level images.

The typical scale of the present binary images is very small (within a range of 12 pixels), and therefore the contribution of structuring elements of large size decreases, whatever the considered training.

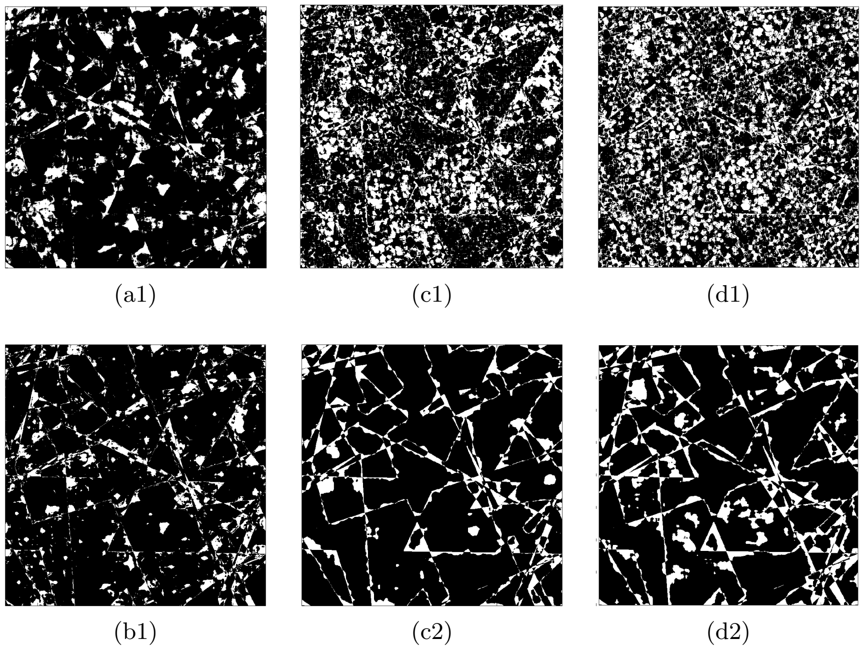
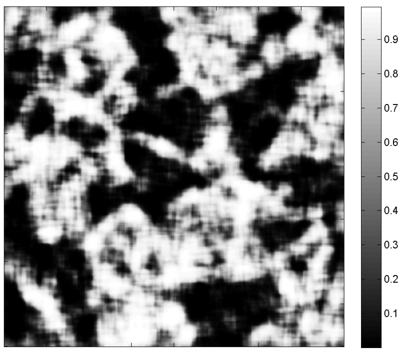


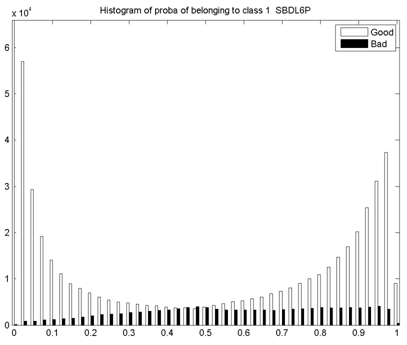
Figure 5. Localisation of misclassified pixels are plotted in white for the 4 studied cases. a, b , c and d correspond to the label in Figure 2. (a1, b1, c1 and d1) Descriptors are not averaged. (c2 and d2) Descriptors are averaged by a 35 × 35 box.

Error localization

The localization of misclassified pixels gives interesting information. For gray scale images (Figure 5(a1) and 5(b1)) errors are mainly located in small areas of the patchwork image and near the boundaries between the two textures. For binary images, this Figure illustrates the differences existing between non averaged descriptors (Figure 5(c1) and 5(d1)) and averaged descriptors (Figure 5(c2) and 5(d2)). The improvement of the descriptors averaging is clearly visible: for the training 1 (Figure 5(c1) and 5(d1)) large areas of identical texture are misclassified, and the error image appears very noisy. On the contrary, for the training 2 (Figure 5(c2) and 5(d2)), the localization of misclassified pixels are mainly located on the boundary between the two textures.



(a)



(b)

Figure 6. (a) Classification probability for image 2(d). (b) The corresponding histogram. In black, pixels that are misclassified and in white, the well classified ones.

We plot in Figure 6 the classification probability for image 2(d) and the corresponding histogram. The probability image could be used as a seed for further segmentation, using for instance watersheds. However, our attempts and a comparison to image 2(d) shows that it would be very difficult to recover the misclassified pixels in small area without introducing errors in the well classified pixels. Therefore the present classification is closed to the optimal one.

5. Conclusion

A morphological approach at the level of pixels in images, combined with statistical learning proved to be very efficient for the segmentation of binary or digital textures, as illustrated for difficult case studies. The main conclusions for further applications are as follows: the basic operations of mathematical morphology (erosion/dilation for binary images, and opening/closing for gray level images) provide efficient descriptors of textures, even on a pixel scale. Local averaging of the binary descriptors improves significantly their discriminant power and an optimal size of window is found for the classification. Finally, the training procedure has to be made on a selection of pixels in composite images, rather than in pure textures in order to minimize the edge effect resulting from boundaries separating the two textures.

References

[1] A. Aubert, D. Jeulin, and R. Hashimoto, *Surface texture classification from morphological transformations*, Proc. ISMM'2000, Mathematical morphology and its applications to Image and Signal Processing, 2000, pp. 253–252.

[2] E. J. Candes and D. L. Donoho, *Curvelets - a surprisingly effective non-adaptive representation for objects with edges*, A Cohen,C Rabut,L L Schumaker Eds. Curve and Surface Fitting, 1999.

[3] F. S. Cohen, Z. Fan, and S. Attali, *Automated inspection of textile fabrics using textural models*, IEEE Transactions on Pattern Analysis and Machine Intelligence **13** (1991), no. 8, 803–808.

[4] M. Elad, J. L. Querre, P. Starckand, and D. Donoho, *Simultaneous cartoon and texture image inpainting using morphological component analysis (mca)*, ACHA (to appear).

[5] G. Fricout and D. Jeulin, *Analisis de imagen y morfologia matematica para la caracterizacion en linea del aspecto de las superficies pintadas*, Pinturas y Acabados **XLVI** (2004), no. 292, 6–13.

[6] J. Iivarinen, *Surface defect detection with histogram-based texture features*, Intelligent robots and computer vision xix: Algorithms, techniques, and active vision, 2000, pp. 140–145.

[7] D. Jeulin, *Multivariate random image models*, Acta Stereologica; 11/SUPPL I (1992), 59–66.

[8] ______, *Random texture models for materials structures*, Statistics and Computing **10** (2000), 121–131.

[9] D. A. Karras and B. G. Mertzios, *Improved defect detection using novel wavelet feature extraction involving principal component analysis and neural network techniques*, Australian joint conference on artificial intelligence, 2002, pp. 638–647.

[10] A. Kumar and G. K. H. Pang, *Defect detection in textured materials using gabor filters*, Industry Applications **38** (2002), no. 2, 425–440.

[11] G. Lambert and F. Bock, *Wavelet methods for texture defect detection*, Proceedings of the 1997 international conference on image processing, 1997, pp. 201–204.

[12] J. Malik, S. Belongie, J. Shi, and T. Leung, *Textons, contours and regions: Cue integration in image segmentation*, Seventh international conference on computer vision (iccv'99), 1999, pp. vol. 2, p. 918.

[13] G. Matheron, *Random sets and integral geometry*, J. Wiley, 1975.

[14] A. Serdaroglu, A. Ertuzun, and A. Ercil, *Defect detection in textile fabric images using wavelet transforms and independent component analysis*, Journal Pattern Recognition and Image Analysis **16** (2006), no. 1, 61–64.

[15] J. Serra, *Image analysis and mathematical morphology*, Academic Press, London, 1982.

[16] K. Sivakumar and J. Goutsias, *Discrete morphological size distributions and densities: estimation techniques and applications*, Journal of Electronic Imaging **6** (1997), no. 1, 31–53.

[17] X. Xie and M. Mirmehdi, *Texture exemplars for defect detection on random textures*, Icapr, 2005, pp. 404–413.

On distances, paths and connections for hyperspectral image segmentation

Guillaume Noyel, Jesus Angulo and Dominique Jeulin

Centre de Morphologie Mathématique (CMM), Ecole des Mines de Paris, Fontainebleau, France
{guillaume.noyel,jesus.angulo,dominique.jeulin}@ensmp.fr

Abstract The present paper introduces the η and μ connections in order to add regional information on λ-flat zones, which only take into account a local information. A top-down approach is considered. First λ-flat zones are built in a way leading to a sub-segmentation. Then a finer segmentation is obtained by computing η-bounded regions and μ-geodesic balls inside the λ-flat zones. The proposed algorithms for the construction of new partitions are based on queues with an ordered selection of seeds using the cumulative distance. η-bounded regions offers a control on the variations of amplitude in the class from a point, called center, and μ-geodesic balls controls the "size" of the class. These results are applied to hyperspectral images.

Keywords: hyperspectral image, connection, quasi-flat zones, η-bounded regions, μ-geodesic balls, top-down aggregation.

1. Introduction

The aim of this paper is to extend and to improve quasi-flat zones-based segmentation. We focus on hyperspectral images to illustrate our developments.

Flat zones, and its generalization, quasi-flat zones or λ-flat zones, initially introduced for scalar functions (i.e., gray-level images) [6] were generalized to color (and multivariate) images [12] (see Definition 4 in Section 2). Classically, λ-flat zones are used as a method to obtain a first image partition (i.e., fine partition). The inconvenience is that the partition typically presents small regions in zones of high gradient (e.g., close to the contours, textured regions, etc.). Several previous works proposed methods to solve it. In [3], Brunner and Soille proposed a method to solve the over-segmentation of quasi-flat zones. They use an iterative algorithm, on hyperspectral images, based on seeds of area larger than a threshold. Their approach consists in computing an over-segmentation and then merging small regions. Besides, in [8], Salembier et al. propose a method to suppress flat regions with area below a given size. They work on the adjacency graph of the flat zones and they define a merging order and criterion between the regions. Moreover, a flat zone, below a given size, cannot be shared by various flat zones. Crespo et al. [4] proposed a similar approach for region merging.

However, the main drawback of λ-flat zones for space partition is that they are very sensitive to small variations of the parameter λ. For instance in the example of Figure 1, we have only 21 or 1 λ-flat zones depending on a slight variation of λ. In fact, the problem of λ-flat zones lies in their local definition: no regional information is taken into account. For the example of Figure 1, with $\lambda = 10$ is obtained only one connected region, presenting a *local smooth variation* but involving a considerable *regional rough variation*. This problem is more difficult to tackle than the suppression of small regions.

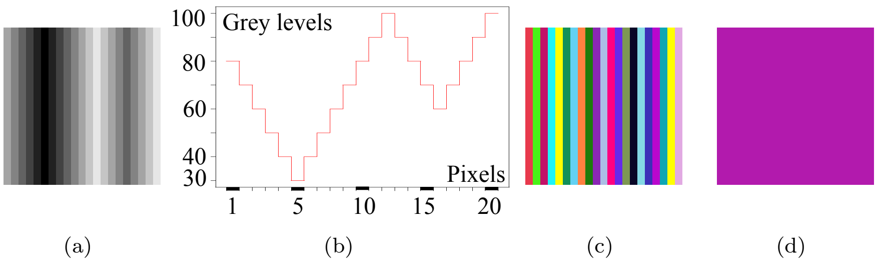


Figure 1. λ-flat zones of image "tooth saw" (21×21 pixels). (a) Image. (b) Row profile. (c) $\lambda = 9.9$ (21 zones). (d) $\lambda = 10$ (1 zone).

The purpose of our study is just to address this issue. We start with an initial partition by λ-flat zones, with a non critical high value of λ, that leads to a sub-segmentation (i.e., large classes in the partition). Then, for each class, we would like to define a second segmentation according to a regional criterion. In fact, two new connections are introduced: (1) η-bounded regions, and (2) μ-geodesic balls; the corresponding algorithms are founded on seed-based region growing inside the λ-flat zones. We show that the obtained reliable segmentations are less critical with respect to the choice of parameters and that these new segmentation approaches are appropriate for hyperspectral images. From a more theoretical viewpoint, the Serra's theory of segmentation [10] allows us to explain many notions which are considered in this paper.

2. General notions

In this section some notions necessary for the sequel are reminded.

Hyperspectral images are multivariate discrete functions with typically several tens or hundreds of spectral bands. In a formal way, each pixel of an hyperspectral image is a vector with values in wavelength, in time, or associated with any index j. To each wavelength, time or index corresponds an image in two dimensions, called channel.

The number of channels depends on the nature of the specific problem under study (satellite imaging, spectroscopic images, temporal series, etc.).

Definition 1 (Hyperspectral image)**.** Let $\mathbf{f}_\lambda : E \to \mathcal{T}^L$ $(x \to \mathbf{f}_\lambda(x) = (f_{\lambda_1}(x), f_{\lambda_2}(x), \ldots, f_{\lambda_L}(x)))$, be an hyperspectral image, where: $E \subset \mathbb{R}^2$, $\mathcal{T} \subset \mathbb{R}$ and $\mathcal{T}^L = \mathcal{T} \times \mathcal{T} \times \ldots \times \mathcal{T}$; $x = x_i \setminus i \in \{1, 2, \ldots, P\}$ is the spatial coordinates of a vector pixel $\mathbf{f}_\lambda(x_i)$ (P is the pixels number of E); $f_{\lambda_j} \setminus j \in \{1, 2, \ldots, L\}$ is a channel (L is the channels number); $f_{\lambda_j}(x_i)$ is the value of vector pixel $\mathbf{f}_\lambda(x_i)$ on channel f_{λ_j}. □

Definition 2 (Spectral distance)**.** A spectral distance is a function $d : \mathcal{T}^L \times \mathcal{T}^L \to \mathbb{R}^+$ which verifies the properties: 1) $d(\mathbf{s}_i, \mathbf{s}_j) \geq 0$, 2)$d(\mathbf{s}_i, \mathbf{s}_j) = 0 \Leftrightarrow \mathbf{s}_i = \mathbf{s}_j$, 3)$d(\mathbf{s}_i, \mathbf{s}_j) = d(\mathbf{s}_j, \mathbf{s}_i)$, 4)$d(\mathbf{s}_i, \mathbf{s}_j) \leq d(\mathbf{s}_i, \mathbf{s}_k)+$ $d(\mathbf{s}_k, \mathbf{s}_j)$, for $\mathbf{s}_i$, $\mathbf{s}_j$, $\mathbf{s}_k \in \mathcal{T}^L$. □

Various metrics distance are useful for hyperspectral points. In this paper, the following two are used:

- Euclidean distance: $d_E(\mathbf{f}_\lambda(x), \mathbf{f}_\lambda(y)) = \sqrt{\sum_{j=1}^{L} (f_{\lambda_j}(x) - f_{\lambda_j}(y))^2}$;
- Chi-squared distance: $d_{\chi^2}(\mathbf{f}_\lambda(x_i), \mathbf{f}_\lambda(x_{i'})) = \sqrt{\sum_{j=1}^{L} \frac{N}{f_{.\lambda_j}} \left(\frac{f_{\lambda_j}(x_i)}{f_{x_i.}} - \frac{f_{\lambda_j}(x_{i'})}{f_{x_{i'}.}} \right)^2}$ with $f_{.\lambda_j} = \sum_{i=1}^{P} f_{\lambda_j}(x_i)$, $f_{x_i.} = \sum_{j=1}^{J} f_{\lambda_j}(x_i)$ and $N = \sum_{j=1}^{L} \sum_{i=1}^{P} f_{\lambda_j}(x_i)$.

Definition 3 (Path)**.** A path between two points x and y is a chain of points $(p_0, p_1, \ldots, p_i, \ldots, p_n) \in E$ such as $p_0 = x$ and $p_n = y$, and for all i, (p_i, p_{i+1}) are neighbours. □

Definition 4 (Quasi-flat zones or λ-flat zones)**.** Given a distance $d : \mathcal{T}^L \times \mathcal{T}^L \to \mathbb{R}^+$, two points x, $y \in E$ belongs to the same quasi-flat zone of an hyperspectral image $\mathbf{f}_\lambda$ if and only if there is a path $(p_0, p_1, \ldots, p_n) \in E^n$ such as $p_0 = x$ and $p_n = y$ and, if, for all i, $(p_i, p_{i+1}) \in E^2$ are neighbours and $d\left(\mathbf{f}_\lambda(p_i), \mathbf{f}_\lambda(p_{i+1})\right) \leq \lambda$, with $\lambda \in \mathbb{R}^+$. □

A path $(p_0, p_1, \ldots, p_n)$ in an hyperspectral image can be seen as a graph in which the nodes correspond to the points connected by edges along the path. For all i, the edge between the nodes p_i and p_{i+1} is weighted by $d\left(\mathbf{f}_\lambda(p_i), \mathbf{f}_\lambda(p_{i+1})\right)$.

Definition 5 (Geodesic path)**.** The geodesic path between two points x and y in E is the path of minimum weight. □

This definition means that the sum of distances $\sum_{i=0}^{n} d\left(\mathbf{f}_\lambda(p_i), \mathbf{f}_\lambda(p_{i+1})\right)$, along this path, $(p_0, p_1, \ldots, p_n)$ such as $p_0 = x$ and $p_n = y$, is minimum. It is called the geodesic distance between x and y and noted $d_{geo}(x, y)$.

In order to compute the geodesic path, the Dijkstra's algorithm can be used [5]. Meanwhile we use an algorithm based on hierarchical queues [11].

For the purposes of segmentation, we need to fix some theoretical notions.

Definition 6 (Partition)**.** Let E be an arbitrary set. A partition $\mathcal{D}$ of E is a mapping $x \to D(x)$ from E into $\mathcal{P}(E)$ such that: (i) for all $x \in E$: $x \in D(x)$, (ii) for all $x, y \in E$: $D(x) = D(y)$ or $D(x) \cap D(y) = \emptyset$. $D(x)$ is called the class of the partition of origin x. □

The set of partitions of an arbitrary set E is ordered as follows.

Definition 7 (Order of partitions)**.** A partition $\mathcal{A}$ is said to be finer (resp. coarser) than a partition $\mathcal{B}$, $\mathcal{A} \leq \mathcal{B}$ (resp. $\mathcal{A} \geq \mathcal{B}$), when each class of $\mathcal{A}$ is included in a class of $\mathcal{B}$. □

This leads to the notion of ordered hierarchy of partitions $\Pi_{i=1}^{N}\mathcal{D}_i$, such that $\mathcal{D}_i \leq \mathcal{D}_{i+1}$, and even to a complete lattice [10].

Definition 8 (Connection)**.** Let E be an arbitrary non empty set. We call connected class or connection $\mathcal{C}$ any family in $\mathcal{P}(E)$ such that: (0) $\emptyset \in \mathcal{C}$, (i) for all $x \in E$, $\{x\} \in \mathcal{C}$, (ii) for each family $C_i, i \in I$ in $\mathcal{C}$, $\cap_i C_i \neq \emptyset$ implies $\cup_i C_i \in \mathcal{C}$. Any set C of a connected class $\mathcal{C}$ is said to be connected. □

It is clear that a connection involves a partition, and consequently a segmentation of E. According to [10], more precise notions than connective criteria (which produce segmentations) can be considered in order to formalize the theory, but this is out of the scope of this paper.

In particular, the λ flat zones can be considered as a connection, λ-flat connection, i.e., λFZ is the partition of the image $\mathbf{f}_\lambda$ according to the λ-flat connection.

For multivariate images, where the extrema (i.e., minimum or maximum) are not defined, the vectorial median is a very interesting notion to rank and select the points [1].

Definition 9 (Vectorial median)**.** A vectorial median of a set $R \subset E$ is any value $\mathbf{f}_\lambda(k)$ in the set at point $k \in R$ such as:

$$k = argmin_{p \in R} \sum_{i/x_i \in R} d\left(\mathbf{f}_\lambda(p), \mathbf{f}_\lambda(x_i)\right) = argmin_{p \in R}\delta_R(\mathbf{f}_\lambda(p)). \quad (1)$$

□

In order to compute the vectorial median $\mathbf{f}_\lambda(k)$, $k \in R$, the cumulative distance $\delta_R\left(\mathbf{f}_\lambda(p)\right)$ has to be evaluated for all $p \in R$. Therefore all points p are sorted, in ascending order, on the cumulative distance. The resulting list of ordered points is called ascending ordered list based on the cumulative distance. The first element of the list is the vectorial median (the last element is considered as the anti-median).

3. η-Bounded regions

One of the main idea is to understand that on quasi-flat zones the distance or slope between two neighbouring points must be inferior to the parameter λ. We can establish a comparison with a hiker, from one point who will only deal with the local slope and not on the cumulative difference in altitude on the λ-flat zones. To consider this limitation we propose a new kind of regional zones: η-bounded regions, according to the following connection.

Definition 10 (η-bounded connection)**.** Given an hyperspectral image $\mathbf{f}_\lambda(x)$ and its initial partition based on λ-flat zones, λFZ, where λFZ_i is the connected class i and $R_i \subseteq E$ (with cardinal K) is the set of points p_k, $k = 0, 1, 2, ..., K-1$, that belongs to the class i. Let p_0 be a point of R_i, named the center of class i, and let $\eta \in \mathbb{R}^+$ be a positive value. A point p_k belongs to the η-connected component centered at p_0, denoted $\eta BR_i^{p_0}$, if and only if $d(\mathbf{f}_\lambda(p_0), \mathbf{f}_\lambda(p_k)) \leq \eta$ and p_0 and p_k are connected. $\square$

For each class λFZ_i the method is iterated using different centers p_j ($j = 0, 1, \cdots J$) until all the space of the λ-connected component is segmented: $\cup_{j=0}^{J} \eta BR_i^{p_j} = \lambda FZ_i$, $\cap_{j=0}^{J} \eta BR_i^{p_j} = \emptyset$, where the η-bounded regions are also connected. Each center p_j belongs to $\lambda FZ_i \setminus \cup_{l=0}^{j-1} \eta BR_i^{p_l}$.

The new image partition associated to η-bounded connection is denoted ηBR. It is evident that this second-class connection is contained in the λFZ initial connection, i.e., $\eta BR(x) \leq \lambda FZ(x)$. As shown by Serra [10], a center or seed p_j is needed to guarantee the connectiveness, thus being precise, the region is bounded with respect to the center. The η-connection is a generalization of the jump connection [9] with the difference that working on hyperspectral images, the seeds p_j cannot be the minima or maxima. We propose to compute the median value as initial seed.

Note that the method can be also applied on the space E of the initial image $\mathbf{f}_\lambda$, without considering an initial λFZ (which is equivalent to take a value of λ equal to the maximal image distance range). The advantage of our approach is that we have now a control of the local variation, limited by λ, and the regional variation, bounded by η. Moreover, the computation of seeds is more coherent when working on relatively homogenous regions.

In practice, for all the K points on each λFZ_i, the ascending ordered list based on cumulative distance δ_i is computed. Then, the first element of the list, i.e., the vectorial median, is used as first seed k of the η-bounded region ηBR_i^1. The distance, from the seed to each point $p \in \lambda FZ_i$, such as $p \in Neigh(q)$, $q \in \eta BR_i^1$, is measured. If this distance is less than η then p is added to ηBR_i^1 and removed from δ_i. For all the others points $q \in \lambda FZ_i \setminus \eta BR_i^1$, the first point of the list δ_i is the seed of the second η-bounded region ηBR_i^2. Then we iterate the process until all the points of the λ-flat zone λFZ_i are in an η-bounded region.

Algorithm 1 μ-geodesic balls.

```
Given a distance $d$, the $\lambda$-flat zones, $\lambda FZ$, of an image $\mathbf{f}_\lambda$, $\delta$ a list of cumulative
distance, $Q$ a queue, $imOut$ an output scalar image
Initialize the value of $\eta$
$currentlabel \leftarrow 0$
for all $\lambda FZ \in \mathbf{f}_\lambda$ do
   for all point $p \in \lambda FZ$ do
      $distance \leftarrow \sum_{q \in \lambda FZ} d(p, q)$
      $\delta \leftarrow$ add the pair $(p, distance)$
   end for
   Ascending sort on parameter $distance$ of $\delta$
   while $\delta$ is not empty do
      $k \leftarrow$ first point of $\delta$
      $Q \leftarrow push(k)$
      while $Q$ is not empty do
         $p \leftarrow pop(Q)$
         $imOut(p) \leftarrow currentlabel$
         Remove $q$ and its $distance$ in $\delta$
         for all $q \in Neigh(p)$ and $q \in \delta$ do
            if $d(k, q) \leq \eta$ then
               $Q \leftarrow push(q)$
            end if
         end for
      end while
      $currentlabel \leftarrow currentlabel + 1$
   end while
end for
```

Using again the mountain comparison, this can be compared to a hiker starting from a point with a walk restricted to a ball of diameter $2 \times \eta$ centered on the starting point. He cannot go upper or lower than this boundary. The points to be reached by the hiker, given the previous conditions, are connected and constitute an η-bounded region.

To understand the effect of these regions, they are applied to the image "tooth saw" with an Euclidean distance d_E (Figure 2). For the sake of simplicity, in this image only the grey levels of the first channel have a shape of "tooth saw" (Figure 1(b)), the others being constant.

We notice from the figure that the smallest the parameter η, the smallest the area of η-bounded regions. Besides, η-bounded regions are very sensitive to the peaks when working on scalar functions. In fact, if η is less than the difference of altitude between the seed and the peak, the points in between are in the same η-bounded region. However, if η is larger than the difference of altitude between the seed and the peak, the points behind the peak, for which the difference of altitude from the seed is less than η, are in the same η-bounded region (Figure 3).

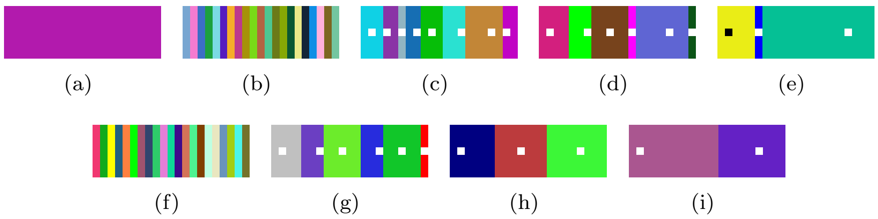


Figure 2. η-bounded regions and μ-geodesic balls of image "tooth saw" ($21 \times 21 \times 4$ pixels) for $\lambda = 10$. Seeds are marked with a white (or black) point, when they are not trivial. (a) λFZ. $\lambda = 10$. (b) $\eta = 0$. (c) $\eta = 10$. (d) $\eta = 20$. (e) $\eta = 30$. (f) $\mu = 0$. (g) $\mu = 20$. (h) $\mu = 40$. (i) $\mu = 100$.

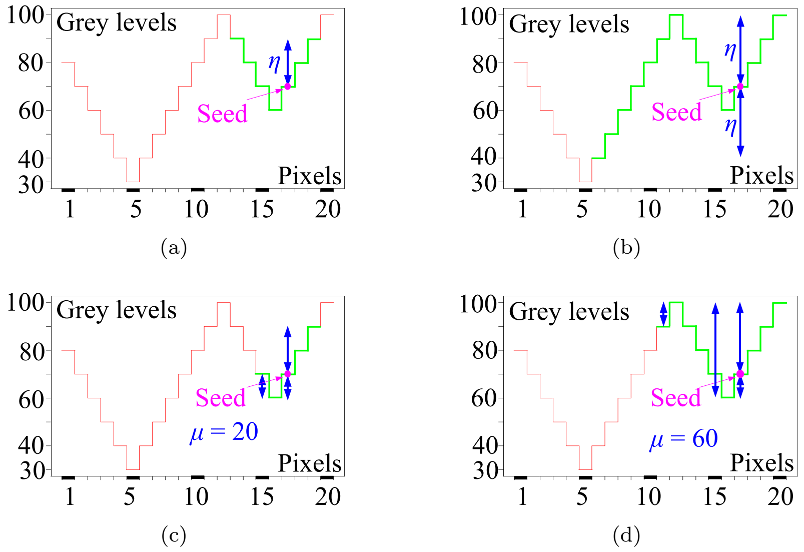


Figure 3. η-bounded region and μ-geodesic ball (in green) on the profile "tooth saw" for $\lambda = 10$. (a) $\eta = 20$. (b) $\eta = 30$. (c) $\mu = 20$. (d) $\mu = 60$.

4. μ-Geodesic balls

As for η-bounded regions, we have created μ-geodesic balls, μGB, to improve the λ-flat zones, but now introducing a control of the dimension of the zone. First of all, λ-flat zones are built. Then, the cumulative difference in altitude is measured from a starting point, the seed, in each λ-flat zone. From this point a geodesic ball of radius μ is computed. This zone is a μ-geodesic ball. It corresponds to the maximum of steps (points) that can be reached by a hiker starting from the seed inside a λ-flat zone, for a given cumulative altitude. μ-geodesic balls are defined using the following connection.

Definition 11 (μ-geodesic connection)**.** Given an hyperspectral image $\mathbf{f}_\lambda(x)$ and its initial partition based on λ-flat zones, λFZ, where λFZ_i is the connected class i and $R_i \subseteq E$ (with cardinal K) is the set of points p_k, $k = 0, 1, 2, ..., K-1$, that belongs to the class i. Let p_0 be a point of R_i, named the center of class i, and let $\eta \in \mathbb{R}^+$ be a positive value. A point p_k belongs to the μ-connected component centered at p_0, denoted $\mu GB_i^{p_0}$ if and only if $d_{geo}(\mathbf{f}_\lambda(p_0), \mathbf{f}_\lambda(p_k)) \leq \mu$. □

It is important to notice, that the geodesic paths imposed the connectivity to the μ-geodesic ball. Formally, for each class λFZ_i the method is iterated using different centers p_j $(j = 0, 1, \cdots J)$ until the full segmentation of the λ-connected component.

Algorithm 2 μ-geodesic balls.

```
Given a distance $d$, the $\lambda$-flat zones, $\lambda FZ$, of an image $\mathbf{f}_\lambda$, $\delta$ a list of cumulative
distance, $Q$ a queue, $imOut$ an output scalar image
Initialize the value of $\mu$
$currentlabel \leftarrow 0$
for all $\lambda FZ \in \mathbf{f}_\lambda$ do
   for all point $p \in \lambda FZ$ do
      $distance \leftarrow \sum_{q \in \lambda FZ} d(p, q)$
      $\delta \leftarrow$ add the pair $(p, distance)$
   end for
   Ascending sort on parameter $distance$ of $\delta$
   while $\delta$ is not empty do
      $k \leftarrow$ first point of $\delta$
      for all point $p$ inside the geodesic ball of center $k$ and radius $\mu$ inside the
      $\lambda FZ$ do
         $imOut(p) \leftarrow currentlabel$
         Remove $p$ and its $distance$ in $\delta$
      end for
      $currentlabel \leftarrow currentlabel + 1$
   end while
end for
```

The new image partition associated to μ-geodesic connection is denoted μGB. As for η-bounded connection, this second-class connection is contained in the λFZ initial connection, i.e., $\mu GB(x) \leq \lambda FZ(x)$. The advantage of this approach is that we have now a regional control of the "geodesic size" of the classes by measuring the geodesic distance, limited by μ inside the local variation limited by λ. In practice, μ-geodesic balls are built as η-bounded regions, except that from each seed the geodesic ball is computed inside the λFZ.

μ-geodesic balls are computed with an Euclidean distance d_E for the image "tooth saw" (Figure 2). We notice that the smaller is μ, the smaller the area of μ-geodesic balls is. Besides, these balls are not very sensitive to the peaks. In fact, if μ is less than the difference of altitude between the seed and the peak, the points in between are in the same μ-geodesic ball. However, if μ is larger than the difference of altitude between the seed and the peak, only the points behind the peak, for which the cumulative altitude from the seed is less than μ, are in the same μ-geodesic ball (Figure 3).

5. Results and discussions

In order to illustrate our results on real images, we extracted η-bounded regions and μ-geodesic balls in the image "woman face" of size $45 \times 76 \times 61$ pixels (Figure 4). The channels are acquired between 400 nm and 700 nm with a step of 5 nm. Moreover, to reduce the number of flat zones in this image, a morphological leveling is applied on each channel, with markers obtained by an ASF (Alternate Sequential Filter) of size 1. Then, λ-flat zones are computed. Besides, the computation time in our current implementation with Python is moderate, i.e., a few minutes. However, note that queues algorithms in C++ are very fast.

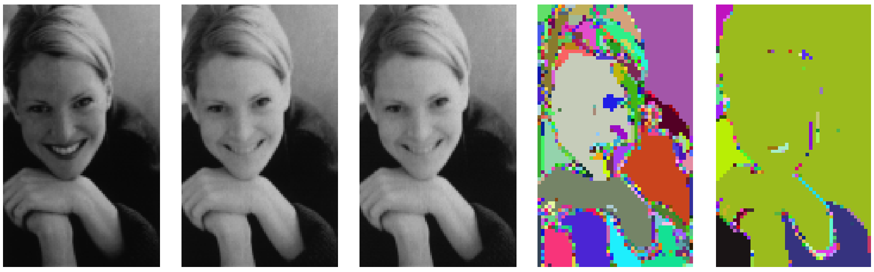

(a) (b) (c) (d) (e)

Figure 4. Channels of image "woman face" $\mathbf{f}_\lambda$ ($45 \times 76 \times 61$ pixels) and λ-flat zones. (a) $\mathbf{f}_{\lambda_{30}}$. (b) $\mathbf{f}_{\lambda_{45}}$. (c) $\mathbf{f}_{\lambda_{61}}$. (d) $\lambda = 0.003$. (e) $\lambda = 0.006$. (Source: Spectral Database, University of Joensuu Color Group, `http://spectral.joensuu.fi/`)

It is important to choose an appropriate distance with respect to the space of the image. In the spectral initial image of "woman face" we choose the Chi-squared distance. For η-bounded regions and μ-geodesic balls, the number of zones minus the number of λ-flat zones is measured. In the Figure 6, we notice that the number of zones decrease with the parameters η or μ. From figures, we notice that η-bounded regions are less sensitive than μ-geodesic balls to small variations on distances between points. However, the area of μ-geodesic balls is more controlled than the area of η-bounded regions. Therefore, η-bounded regions are better to find the details and μ-geodesic balls are better to build smoother zones.

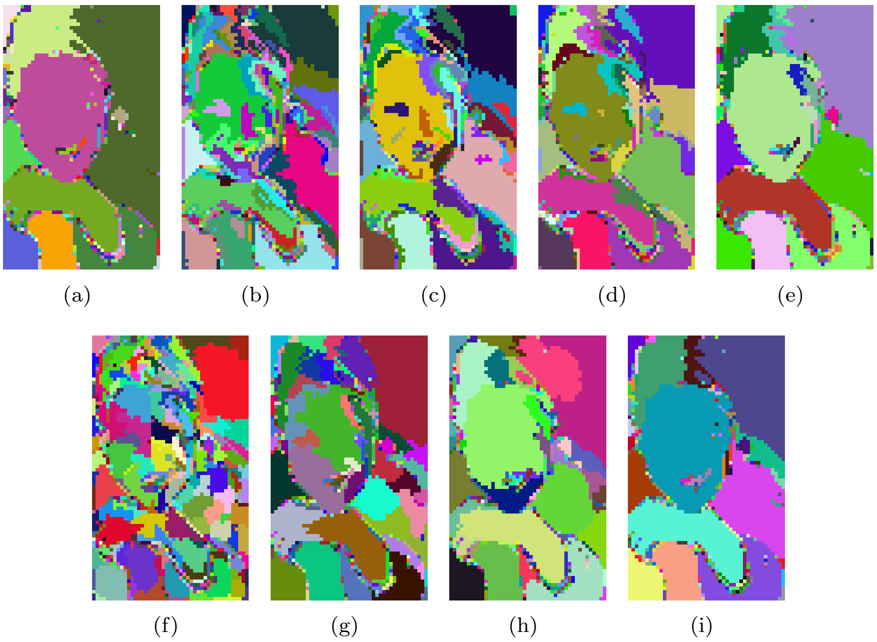


Figure 5. η-bounded regions and μ-geodesic balls of image “woman face” for $\lambda = 0.005$ (Chi-squared distance d_{χ^2}). (a) λFZ. $\lambda = 0.005$. (b) $\eta = 0.007$. (c) $\eta = 0.009$. (d) $\eta = 0.011$. (e) $\eta = 0.02$. (f) $\mu = 0.01$. (g) $\mu = 0.02$. (h) $\mu = 0.03$. (i) $\mu = 0.05$.

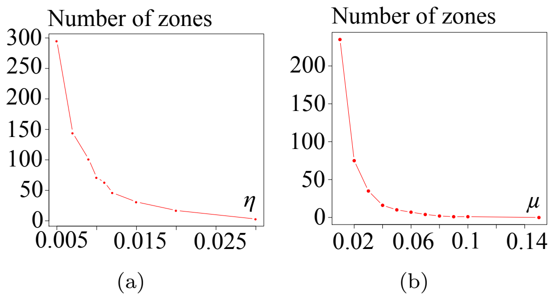


Figure 6. Variations of the number of ηBR or μGB minus the number of λFZ versus the parameter η or μ in image “woman face” for $\lambda = 0.005$ (Chi-squared distance d_{χ^2}). (a) ηBR. (b) μGB.

Besides, in order to evaluate the influence of choosing the vectorial median as a reference seed, we have tested the use of the reverse order for the ascending ordered list based on cumulative distance. In fact, this order corresponds to the vectorial anti-median (Figure 7). Comparing these figure to the zones obtained with the median seed (Figure 5), we notice that almost the same zones are obtained. Consequently, η-bounded regions and μ-geodesic balls have a small dependence to the chosen seeds.

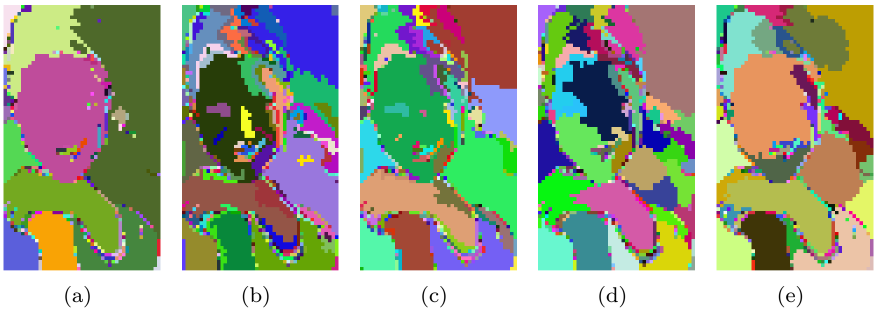


Figure 7. η-bounded regions and μ-geodesic balls with an anti-median seed in image "woman face" for $\lambda = 0.005$ (Chi-squared distance d_{χ^2}). (a) λFZ, $\lambda = 0.005$. (b) $\eta = 0.009$. (c) $\eta = 0.011$. (d) $\mu = 0.02$. (e) $\mu = 0.03$.

Moreover, the same segmentations can be obtained in factor space using an Euclidian distance because it is equivalent to Chi-squared distance in image space. We have also computed it, keeping three factorial axes with relative inertia: 87.2 %, 10.2 % and 1.5 %. By reducing the volume of data, the computation is more efficient on 3 channels than on 61.

6. Conclusion and perspectives

We have presented two new connected zones: η-bounded regions and μ-geodesic balls. They improve the λ-flat zones, which deals only with local information, by introducing regional information. Moreover, these new connections are of second order because they are built, and included, in the λ-flat zones which are already connected. The approach consists in selecting a sufficiently high parameter λ to obtain first a sub-segmentation. Then, η-bounded regions or μ-geodesic balls are built, leading to a segmentation by a top down aggregation. The η-bounded regions introduce a parameter controlling the variations of distance amplitude in the λ-flat zones, meanwhile μ-geodesic balls introduce a parameter to control the size by controlling the cumulative amplitude inside the λ-flat zones.

Besides, these two second order connections produce pyramids of partitions with a decreasing number of regions when the value of η or μ is increased, until the partition associated to the last level is equal to the partition defined by the λ-flat zones. However, it is important to notice that this pyramid is not an ordered hierarchy in the meaning that the classes are not ordered by increasing level.

Furthermore, we have proposed algorithms for the construction of partitions associated to both new types connections, η and μ, which are based on queues with an ordered selection of seeds using the cumulative distance.

About the perspectives, we notice that the proposed method does not solve the problem of small classes of the initial partition of the λ-flat zones. However, we can combine our method with the approaches aggregating smaller regions to these of larger area [3,4,8]. For the future, we are thinking on more advanced methods in order to select the seeds, and to determine locally, for each λ class, the adapted value of η or μ.

References

[1] J. Astola, J. Haavisto, and Y. Neuvo, *Vector Median Filters,* Proc. IEEE Special Issue on Multidimensional Signal Processing, Vol. 78 (4), 1990.

[2] J. P. Benzécri, *L'Analyse Des Données. L'Analyse des Correspondances.*, Vol. 2, Dunod, 1973.

[3] D. Brunner and P. Soille, *Iterative area seeded region growing for multichannel image simplification* (C. Ronse, L. Najman, and E. Decencière, eds.), Springer, 2005, Proc. ISMM'05, International Symposium on Mathematical Morphology, pp. 397–406.

[4] J. Crespo, R. Schafer, J. Serra, C. Gratin, and F. Meyer, *The flat zone approach: a general low-level region merging segmentation method*, Signal Processing 62, 1997, pp. 37–60.

[5] E. W. Dijkstra, *A Note on Two Problems in Connection with Graphs*, Numerische Mathematik, 1959, pp. 269–271.

[6] F. Meyer, *The levelings* (H. Heijmans and J. Roerdink, eds.), Kluwer, 1998, Proc. ISMM'98, International Symposium on Mathematical Morphology, 1998, pp. 199–206.

[7] G. Noyel, J. Angulo, and D. Jeulin, *Morphological Segmentation of hyperspectral images*, submitted to Image Analysis and Stereology, ICS XII St Etienne 30 Août-7 Sept 2007, Internal notes Ecole des Mines de Paris nř N-36/06/MM.

[8] P. Salembier, L. Garrido, and D. Garcia, *Auto-dual connected operators based on iterative merging algorithms* (H. Heijmans and J. Roerdink, eds.), Kluwer, 1998, Proc. ISMM'98, International Symposium on Mathematical Morphology, pp. 183–190.

[9] J. Serra, *Set connections and discrete filtering*, (Proc. DGCI 1999) Lecture Notes in Computer Science (M. Couprie and L. Perroton, eds.), Vol. 1568, Springer, 1999, pp. 191–206.

[10] J. Serra, *A lattice approach to image segmentation*, Springer Science, 2006, Journal of Mathematical Imaging and Vision 24, pp. 83–130.

[11] L. Vincent, *Minimal Path Algorithms for the Robust Detection of Linear Features in Gray Images*, Proc. ISMM'98, International Symposium on Mathematical Morphology, 1998, pp. 331–338.

[12] F. Zanoguera and F. Meyer, *On the implementation of non-separable vector levelings* (H. Talbot and R. Beare, eds.), CSIRO, 2002, Proc. ISMM'02, International Symposium on Mathematical Morphology, pp. 369–377.

Generalized watershed and PDEs for geometric-textural segmentation

Anastasia Sofou and Petros Maragos

School of Electrical and Computer Engineering, National Technical University of Athens, Greece
{sofou,maragos}@cs.ntua.gr

Abstract In this paper we approach the segmentation problem by attempting to incorporate cues such as intensity contrast, region size and texture in the segmentation procedure and derive improved results compared to using individual cues separately. We propose efficient simplification operators and feature extraction schemes, capable of quantifying important characteristics like geometrical complexity, rate of change in local contrast variations and orientation, that eventually favor the final segmentation result. Based on the morphological paradigm of watershed transform we investigate and extend its Partial Differential Equation (PDE) formulation in order to satisfy various flooding criteria, and couple them with texture information thus making it applicable to a wider range of images.

Keywords: watershed, flooding, multi-cue segmentation, PDE.

1. Introduction

In this work we treat image segmentation as a set of procedures that need to be followed starting from the initial image and yielding the final partitioning perceived either as a region map or a segmentation boundary. Independently of the method used to achieve the partitioning, this can be divided into the following stages: (i) *image simplification* (ii) *feature extraction* and (iii) *partitioning* into disjoint regions. The simplification stage encompasses tasks such as smoothing, noise reduction, redundant information removal (resulting in an image consisting mostly of flat and large regions), as well as image decomposition into constituent parts. The feature extraction deals with gradient features computation, texture measurements, marker extraction, whereas the final stage of partitioning is the application of the selected segmentation algorithm so as to produce a region map of the image.

Motivated by the efficacy of watershed transform along with latest trends in image segmentation research that encourage combination of different cues [2, 9], we try to incorporate the generalized flooding concept of watershed, thus exploiting intensity contrast and region size criteria [19], with other perceptually meaningful image characteristics such as texture, aiming at improved segmentation results. Additionally, we aim at integrating the aforementioned ideas with Partial Differential Equation (PDE) modeling.

In this paper we propose well-motivated and efficient image simplification and feature extraction techniques as necessary tasks of the presegmentation part. We focus on generalized watershed techniques, investigate their PDE formulation encompassing various flooding criteria, such as region size and volume, and incorporate geometric and textural features in flooding using a leveling-based image decomposition scheme. The resulting segmentation method couples contrast, size and texture information driven by two separate image components: Cartoon U (for contrast information) and Texture component V, resulting from $U + V$ image decomposition model. The modeling is done via PDEs using ideas from curve evolution and level sets, and the implementation is accomplished by adapting specialized level set methodologies, which ensure speed and reduced computational cost. The performance and efficacy of the proposed segmentation scheme is demonstrated through a set of qualitative, quantitative and comparative experimental results.

2. Image simplification

The simplification stage is concerned with noise and redundant information removal, resulting in an image with smoother structure, but at the same time with key features accurately preserved, easier to handle and more appropriate for further processing such as feature extraction and partitioning. The primary concern here is the selection of the filtering the image has to undergo in order to retain meaningful information but at the same time suppress pointless structures without causing boundary blurring or contour displacement. An efficient family of filters that have the aforementioned properties are the morphological connected operators [11,12,17]. For image simplification we use contrast/area/volume filtering and levelings. Further, generalized openings γ and closings φ are often combined sequentially to produce Alternating Sequential Filters (ASF): $\Psi_{\text{ASF}}(I) = \varphi_n\gamma_n \cdots \varphi_2\gamma_2\varphi_1\gamma_1(I)$, where $i = 1, 2, \ldots n$ denotes the increasing scale of the filter.

Contrast filtering The graylevel reconstruction opening ρ^- and closing ρ^+ of an image $I(x, y)$ given a marker signal $M(x, y)$ are:

$$\rho^-(M|I) = \lim_{n\to\infty} F_n,\ F_n = \delta_B(F_{n-1}|I),\ F_0 = M, \tag{1}$$

$$\rho^+(M|I) = \lim_{n\to\infty} F_n,\ \ F_n = \varepsilon_B(F_{n-1}|I),\ F_0 = M, \tag{2}$$

where $\delta_B(M|I)$ and $\varepsilon_B(M|I)$ denote the conditional dilation and erosion, respectively, of M by a unit disk B constrained by I. To achieve contrast filtering we set the marker $M = I - h$ and $M = I + h$ for reconstruction opening and closing, respectively, with h being a constant that controls the contrast of the bright/dark connected components that will be merged.

Self-dual filtering The above operators are either anti-extensive or extensive, simplifying bright or dark image components, respectively. Symmetrical simplification of image components requires self-dual filters, such as the **levelings**, which are nonlinear, increasing and idempotent filters that have many interesting scale-space properties [11, 12]. They treat symmetrically the image foreground and background; further, they can be analyzed as composition of reconstruction opening and closing. They operate on a reference image I by locally expanding/shrinking an initial seed image, called the marker M, and globally constraining the marker evolution by the reference image. Specifically, iterations of the image operator $\lambda(F|I) = (\delta(F) \wedge I) \vee \varepsilon(F)$, where $\delta(F)$ (resp. $\varepsilon(F)$) is a dilation (resp. erosion) of F by a small disk, yield in the limit the leveling of I w.r.t. M,

$$\Lambda(M|I) = \lim_{k\to\infty} F_k, \; F_k = \lambda(F_{k-1}|I), \; F_0 = M. \tag{3}$$

Levelings preserve the coupling and sense of variation in neighbor image values and do not create any new regional maxima or minima across scales. In practice, they can reconstruct whole image objects with exact preservation of their boundaries and edges. In this reconstruction process they simplify the original image by completely eliminating smaller objects inside which the marker cannot fit.

Area filtering The need often occurs to filter out small light (respectively dark) particles from graylevel images without damaging the remaining structures. The operator that achieves this kind of filtering is the area opening (closing) of size n that keeps only the light (dark) connected components whose area (number of pixels) is equal or greater than a threshold n. For binary images $X = \bigcup_i X_i$ expressed as disjoint union of connected components X_i, the area opening is $\alpha_n^-(X) = \bigcup\{X_j \; : \; \text{Area}(X_j) \geq n\}$. Dually, the binary area closing is $\alpha_n^+(X) = [\alpha_n^-(X^c)]^c$. The graylevel area opening is defined via threshold superposition:

$$\alpha_n^-(I)(x,y) = \sup\{h \; : (x,y) \in \alpha_n^-(T_h(I))\}, \tag{4}$$

where $T_h(I) = \{(x,y) : \; I(x,y) \geq h\}$ are the upper level sets of the image I by thresholding it at level h. Similarly for the graylevel closing.

Volume filtering A combination of the above contrast and size connected operators yields the volume reconstruction operator. Volume operators remove connected components from the image whose volume is below a certain threshold. They are defined as:

$$\beta_n^-(I)(x,y) = \sup\{h \; : (x,y) \in \beta_n^-(T_h(I))\}, \tag{5}$$

$$\beta_n^+(I)(x,y) = \sup\{h \; : (x,y) \in \beta_n^+(T_h(I))\}, \tag{6}$$

where in the binary case, if $T_h(I) = X = \bigcup_i X_i$ we define $\beta_n^-(X) = \bigcup\{X_j : \text{Area}(X_j) \cdot h \geq n\}$ with X_j being connected components. Volume operators present the formal properties of openings and closings and can be used as a mean of simplification filtering that balances contrast and size criteria.

3. Feature extraction

The feature extraction stage deals with the extraction of special image features which facilitate the final segmentation step, and requires a more severe but detailed processing of the image. As *features* we denote regions of interest, gradients, texture measurements, as described below.

Gradient features High values of the image's gradient are indicative of abrupt intensity changes and specify possible object/region contours. Additionally, the topographic relief, emerging from the gradient magnitude function is used in the flooding process that leads to the final segmentation map. There are many different types of gradients that have been extensively used in the edge detection framework. Among them, we choose the morphological gradient $\mathcal{M}_{\nabla}(I) = [(I \oplus B) - (I \ominus B)]/2$ for its robust behavior low complexity, and better segmentation results compared to other edge strength operators.

Texture features A way of acquiring texture information from an image I is via a decomposition scheme [13, 21], according to which the image is expressed as $I = U + V$, where U is the "cartoon component" and consists of relatively flat plateaus for the object regions surrounded by abrupt edges, whereas V is the "texture oscillation" and contains texture information plus noise. Simple texture patterns appearing in V component can be modeled as narrowband 2D AM-FM signals [4, 7] of the form $\alpha(x, y)\cos[\varphi(x, y)]$, with a spatially varying amplitude $a(x, y)$ and a spatially-varying instantaneous frequency vector $\vec{\omega}(x, y) = \nabla\varphi(x, y)$ The amplitude is used to model local image contrast and the frequency vector contains rich information about the locally emergent spatial frequencies. An efficient way to estimate the 2D amplitude and frequency signals is via the 2D Teager energy operator [7] $\Psi(f) \triangleq ||\nabla f||^2 - f\nabla^2 f$. Applying Ψ to the AM-FM signal yields $\Psi[a\cos(\varphi)] \approx a^2||\vec{\omega}||^2$, i.e., the product of the instantaneous amplitude and frequency magnitude squared, which may be called the *texture modulation energy.* Complex (wideband) image textures can be modeled as a sum of 2D AM-FM signals; i.e., $f(x, y) = \sum_{k=1} \alpha_k(x, y)\cos[\varphi_k(x, y)]$. In our case, Ψ is applied on narrowband versions of the wideband signal V, which are obtained by convolving it with a dense filterbank [4] of 2D Gabor filters h_k. The modulation energies of the filtered texture components are measured via the 2D Energy Operator Ψ, smoothed by a local averaging filter h_a and then are subjected to pixelwise comparisons. This yields the *Maximum Average Teager Energy* $\Psi_{\text{mat}}[f(x, y)] = \max_k h_a * \Psi\,[f * h_k]\,(x, y)$. It is a slowly-varying indication of texture modulation energy, which can classify among different energy levels. It provides both local and global texture information and tracks the most dominant texture components along multiple modulation bands [5]. The derived image texture feature is capable of quantifying important characteristics like geometrical complexity, rate of change in local contrast variations and texture scale.

Markers Markers are predefined image locations that serve as starting points of the region-growing procedure. These seed points grow in time according to a set of specified criteria until the image plane is totally covered by them. It is common practice that markers are chosen as regions where some homogeneity criterion is constant or a key characteristic is of certain strength. In our research work we emphasize on contrast, volume and texture-based markers, i.e., image areas where the homogeneity criterion is contrast, volume (area and contrast) and texture, respectively. In all three cases we extract markers via a reconstruction procedure as valleys or peaks of an image transform that resembles one of the aforementioned characteristics. In all cases, the scale is incorporated in the structuring element or reconstruction controlling parameter. Specifically, we distinguish the following cases.

Contrast, Area or Volume - based markers. Markers are estimated as valleys (or peaks) of certain strength of a generalized Bottom (Top) Hat Transform. The Bottom Hat Transform is defined as: $H_B(I) = \varphi(I) - I$, where $\varphi(I)$ is a generalized closing and I is an intensity image (initial or simplified). Similarly, Top Hat Transform is defined as: $H_T(I) = I - \gamma(I)$, where $\gamma(f)$ is a generalized opening. Depending on what kind of closing /opening transform we choose, we obtain: (a) *contrast* markers if the generalized closing is based on reconstruction, i.e., $\varphi(I) = \rho^+(I + h|I)$ that is where the parameter h controls the contrast (valley depth), (b) *area* markers if $\varphi(I)$ is area closing, (c) *volume* markers if $\varphi(I)$ is volume closing, in which case contrast and area criteria are exploited.

Texture - based markers. Again markers are estimated as peaks of an image transform that relies on texture characteristics. Therefore, peaks (valleys) either of the texture component V or its dominant modulation energy are extracted as highly (poorly) textured regions. The peak (valley) extraction is based on a reconstruction procedure as discussed earlier.

4. Generalized watershed and PDEs

Apart from the standard morphological flooding approach implemented either via immersion simulations [22] or hierarchical queues [1], the watershed transform has also been modeled in a continuous way via the eikonal PDE [14] and implemented in [8] using curve evolution and level sets. Further, generalized floodings and corresponding watersheds have been investigated in [10]. Using a PDE-based modeling in the flooding process of watershed transform, each emanating wave's boundary is viewed as a curve, which evolves with predefined speed. In the case of uniform height watershed flooding, let us consider a moving smooth closed curve, which is the boundary of the marker region, $\vec{C}(p, t)$ where $p \in [0, 1]$ parameterizes the curve and t is an artificial marching parameter. The PDE that implements the generalized watershed flooding is:

$$\frac{\partial \vec{C}}{\partial t} = \frac{c}{\text{Area}(t)\|\nabla I\|} \cdot \vec{N}, \tag{7}$$

where c is a constant, $\|\nabla I\|$ is the gradient magnitude of the image function I, $\vec{N}$ is the unit outward vector normal to the curve, and Area(t) is either 1 if we perform only contrast-based segmentation (height flooding) or Area(t) = Area($\vec{C}$), that is Area(t) is equal to to area enclosed by the propagating curve at the specific time t in case of contrast and size segmentation (volume flooding) [19]. The above propagation PDE implies that the evolution speed is inversely proportional to the intensity (volume) variation at each image point, in the direction of the outward normal vector. For implementation we use the level set approach [15] where at each time the evolving curve is embedded as the zero level set $\Gamma(t) = \{(x, y) : \Phi(x, y, t) = 0\}$ of a higher dimension space-time function $\Phi(x, y, t)$. Then this embedding function Φ evolves in space-time according to the following PDE:

$$\frac{\partial \Phi}{\partial t} = \frac{c}{\text{Area}(t)\|\nabla I(x,y)\|}\|\nabla \Phi\|. \tag{8}$$

Modeling generalized watersheds via the eikonal has the advantage of a more isotropic flooding but it also introduces some challenges in the implementation. Efficient algorithms [18] to solve time-dependent eikonal PDEs are the narrow-band level sets methods, and more specifically, the fast marching method, an algorithm for stationary formulations of eikonal PDEs.

Experimental results using height and volume flooding segmentation of Equation 7 are illustrated in Figure 1, exploiting the basic property of volume flooding, i.e., retaining the balance between area and contrast. The image shown left in Figure 1 is synthetically produced by taking the distance transform of the corresponding binary image and adding an arbitrary constant to each of their connected components. For illustration purposes, a flooding source has been superimposed for each object. Bright objects appear with higher altitude compared to darker objects. Next in Figure 1 we illustrate the contour lines of each object, with blue color corresponding to lower altitude and red corresponding to higher altitude. The cases of uniform height and volume flooding are examined and presented in third and fourth column of Figure 1, respectively. In the case of height flooding the object of lowest contrast is totally lost, whereas in the case of volume flooding the undetected object is the one of lowest volume (area and contrast).

5. Coupled contrast-texture segmentation

The aforementioned generalized watershed segmentation schemes use as prominent characteristic the image intensity viewed either as seeds' contrast, size or volume. Any textural information present in the image is incorporated in intensity. Based on evidence from psychophysics according to

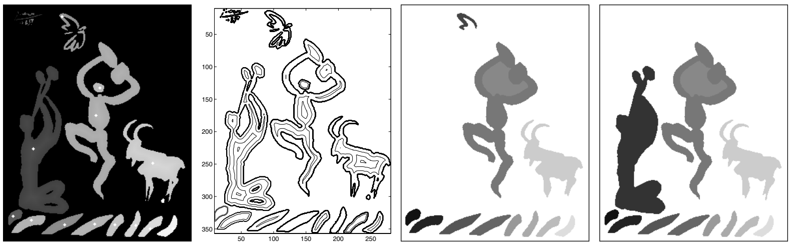


Figure 1. Height and volume flooding segmentation results: (from left to right) synthetic image, contours corresponding to different altitudes (gray values), height flooding segmentation regions, volume flooding segmentation regions.

which humans combine multiple cues in order to detect boundaries from images [16], we try to exploit contrast and texture as two separate information sources so as to improve and balance the segmentation results and eliminate false boundaries introduced by intensity variations in amplitude and phase, owing to textured parts. Ideally we want to add a texture-controlled term to the height/flooding PDE (7) that will be able to quantify properly the available image texture information by enabling the growing seeds surpass false edges introduced by texture structures in the image, thus speeding up the evolution at such places. This can be achieved by the Ψ_{mat} operator, which provides both local and global texture information, tracks the most dominant texture components along multiple modulation bands and is capable of quantifying important characteristics like geometrical complexity, rate of change in local contrast variations and texture scale. We thus conclude to the following PDE:

$$\frac{\partial \vec{C}}{\partial t} = \left(\frac{\lambda_1}{\max(\epsilon, \text{Area}(t)||\nabla I||)} + \lambda_2 \Psi_{\text{mat}}(I) \right) \vec{N}, \tag{9}$$

where λ_1 and λ_1 are parameters that control the contribution of each cue and $0 \leq \epsilon \leq 1$ is used to handle instabilities caused by gradient's zero values. The seeds' evolution speed depends on two eikonal terms, linked with some optimality criterion. The first term drives the curve (seed's boundary) with speed that maximizes the *flooding* of the image toward its watershed. The second term can be shown to correspond to a flow that maximizes the average texture energy: $\max \iint_{R(C)} \Psi(I) \Longrightarrow \partial \vec{C}/\partial t = \Psi(I)\vec{N}$. This term pushes the curve toward regions with large average texture energy.

The PDE (9) consists of two terms: the gradient magnitude operator quantifying intensity changes and the energy modulation operator quantifying AM-FM variations corresponding to texture. Our next concern is to apply these two different operators of separate image transformations emphasizing on different type of information. We take advantage of the recently proposed image decomposition model [13, 21], which provides an effective way of linearly distinguishing contrast and texture from a single image, in the form $I = U + V$. Specifically, the U component, known as *cartoon*, serves very well as a contrast descriptor since it consists of relatively flat plateaus that correspond to object regions, surrounded by abrupt edges that correspond to object boundaries. The V component, which is in fact the *texture oscillation* contains texture oscillations plus noise information and serves as texture descriptor. Combining the $U+V$ image decomposition philosophy with the PDE (9) and level set formulation [15] we derive the following coupled segmentation PDEs:

$$\frac{\partial \vec{C}}{\partial t} = \left(\frac{\lambda_1}{\text{Area}(t)||\nabla U||} + \lambda_2 \Psi_{\text{mat}}(V) \right) \vec{N}, \tag{10}$$

$$\frac{\partial \Phi}{\partial t} = \left(\frac{\lambda_1}{\text{Area}(t)||\nabla U||} + \lambda_2 \Psi_{\text{mat}}(V) \right) ||\nabla \Phi||. \tag{11}$$

Contrast variations are taken into account from the U part, which is obtained by applying the leveling operator on the initial image and texture oscillations are approached through the residual $V = I - U$. The elimination of division-by-zero scenario as introduced in Equation 9 can be as well applied in Equation 10 and Equation 11 in order to handle instabilities.

In the PDE derived above each cue's contribution is controlled by a coefficient, namely λ_1 geometric evolution controlling parameter and λ_2 texture evolution controlling parameter. We set these λ parameters to be spatially adaptable, taking advantage of the fact that the $U + V$ image decomposition model gives evidence about the existence of each component (geometry and texture) at every image location. All the needed information about contrast at each image pixel is encapsulated by $1/|\nabla U|$ component and texture contribution is captured by $\Psi_{\text{mat}}(V)$. Hence, we estimate λ_1 (geometric coefficient) and λ_2 (textural coefficient) as the mean square error between the observed image I and the texture V or contrast U component, respectively. These mean square errors are weighted locally by a small Gaussian window $G_\sigma(x, y)$ of scale σ, i.e., $\lambda_1(x, y) = [G_\sigma * (I - V)^2](x, y)$ and $\lambda_2(x, y) = [G_\sigma * (I - U)^2](x, y)$. We can either use this estimated λ-space functions directly or normalize their sum to 1. Alternatively, the coefficients can be estimated as: $\lambda_1(x, y) = \exp(-[G_\sigma * (I - U)^2](x, y))$ and $\lambda_2(x, y) = \exp(-[G_\sigma * (I - V)^2](x, y))$. The former selection of λ parameters has experimentally been found to yield slightly better results.

The curves that propagate according to the aforementioned evolution scheme are multiple, initialized as the contours of a set of markers, thus indicating significant image regions. The marker extraction is done according to the methodologies described in Section 3. Specifically, depending on the type of image to be segmented we choose our markers to be contrast-oriented, texture-oriented, a combination of the above or manually placed at areas of interest. The PDE (11) is of pure eikonal-type and its implementation is based on established techniques from level sets methods, specifically the fast marching methodology (FMM) [18, 20] that ensures computational speed.

6. Experiments, comparisons, and conclusions

In Figure 2 we demonstrate a set of the extracted features and segmentation results on a biomedical image from prostate tissue used for gleason scale measurement. The reference image is shown left on top row of Figure 2. In same row we illustrate the automatically extracted marker set, U and V image components obtained after image decomposition. In the second row we illustrate the texture modulation energies $\Psi_{\text{mat}}(I)$, and $\Psi_{\text{mat}}(V)$, as well as the corresponding segmentation results using PDEs (9) and (10).

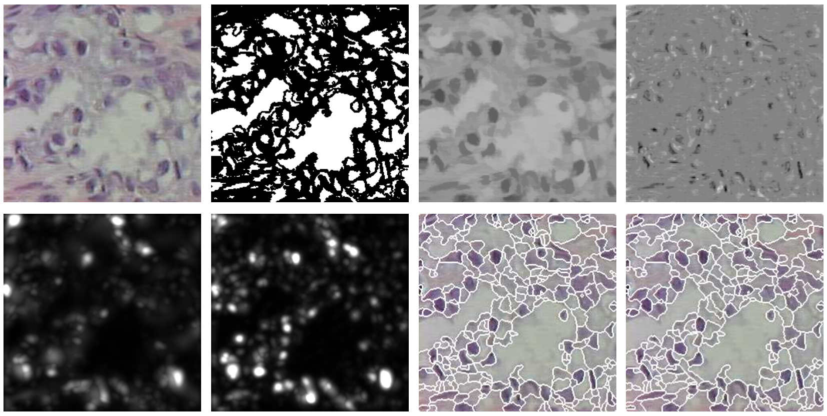


Figure 2. Image features and segmentation results: (from left to right) Original image, Markers, Cartoon U, Texture V, Texture modulation energy $\Psi_{\text{mat}}(I)$, Texture modulation energy $\Psi_{\text{mat}}(V)$, Coupled segmentation on I, Coupled segmentation on $U + V$.

In order to judge the quality of the obtained segmented images, we have used some quality measures in order to quantify the results and test them against other segmentation methodologies. Although there is a variety of goodness criteria for the evaluation of segmentation methodologies, and each criterion can be used in different segmentation scenarios, there is no global measure that can be applied in every case. Among the goodness measures [23] established according to human perception and intuition, we eventually concluded to measure each region intensity variance using a cartoon version of the image, as well as each region's modulation energy variance, thus incorporating both contrast and texture information. The lower those variance values, the better are the segmentation results.

The proposed method was tested against height and volume flooding watershed segmentation, as well as the multicue scheme without image decomposition of Equation 9, since these methods produce similar segmentation results in terms of closed boundaries and disjoint, plane-filling regions.

In Figure 3, we provide a set of different segmentation results obtained by applying the aforementioned methods on four different reference images: 1) a soilsection image consisting of highly contrasted and textured areas, 2) an aerial photo 3) a biomedical image of prostate tissue and 4) an animal image of differently textured areas. The different segmentation methodologies are tested using the same set of automatically extracted markers via contrast or volume criteria for each image (in the case of the animal image markers are placed manually). Marker sets' illustration is omitted due to lack of space. Apart from visual comparisons, we provide Table 1, where the aforementioned goodness measures are computed for each case. As it can be observed, the proposed scheme incorporating image decomposition outperforms the other segmentation methodologies. It provides better results, in the sense that the resulting partitioning map consists of more uniform regions (low cartoon variance values) with smoother texture (low modulation energy variance), compared to the other methodologies.

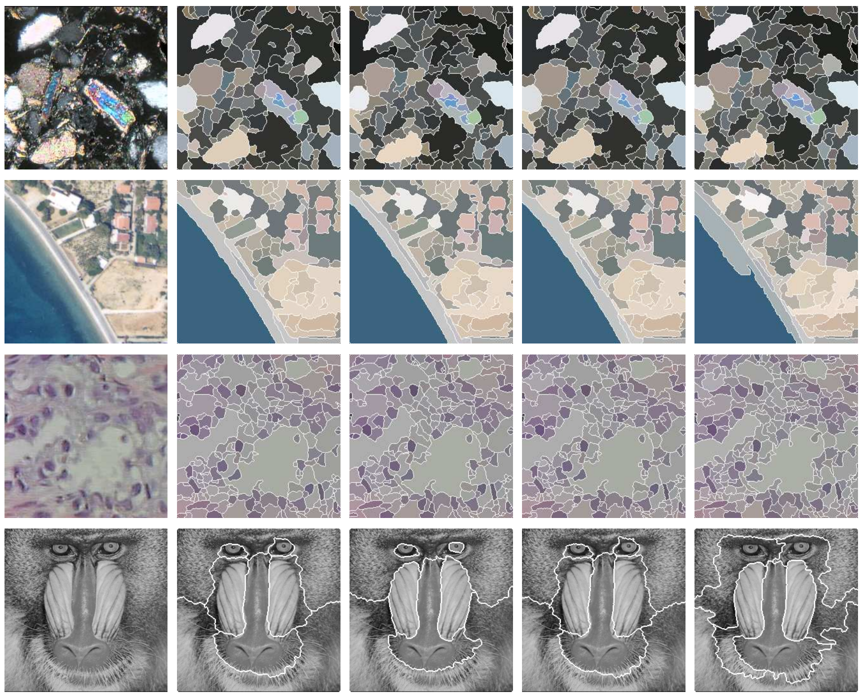


Figure 3. Comparisons of different types of watershed-like segmentation results: (columns from left to right) Reference images, Multicue segmentation results without decomposition, Multicue segmentation results with decomposition, Height watershed flooding segmentation results, Volume watershed flooding segmentation results.

Table 1. Segmentation comparisons.

	Quality Measures	Segmentation Method			
		Coupled Type		Watershed Flooding	
		I	U + V	Height	Volume
soil	var(U)	0.921	0.823	0.893	1.108
	var(Ψ_{mat}(V))	0.280	0.259	0.281	0.254
	length(Γ)	4855	4987	4982	5742
aerial	var(U)	0.335	0.281	0.337	0.383
	var(Ψ_{mat}(V))	0.473	0.468	0.479	0.555
	length(Γ)	3934	4206	4054	4442
biomed	var(U)	0.327	0.294	0.314	0.365
	var(Ψ_{mat}(V))	0.138	0.135	0.140	0.139
	length(Γ)	6529	6630	6728	7593
madrill	var(U)	0.046	0.024	0.046	0.034
	var(Ψ_{mat}(V))	0.272	0.232	0.271	0.285
	length(Γ)	1167	1210	1201	1960

Concluding remarks The presented research work addressed the problem of image segmentation in terms of simplification, feature extraction and image partitioning with focus on a generalized flooding procedure using geometric and textural information. Generalized watershed transform was modeled via PDEs and extended to incorporate geometric and textural information using ideas such as $U+V$ image decomposition and texture AM-FM modeling. The quality of segmentation results was illustrated through qualitative, quantitative and comparative results.

It should be noted that geometric curve evolution of the form $\partial\vec{C}/\partial t = g(c - \mu\kappa)\vec{N}$ has been proposed by Caselles et al [3] and Malladi et al [6]. However, our proposed scheme has three differences compared to the aforementioned evolution: i) it has a term that achieves watershed type flooding ii) it has a second term that is a new contribution and acts on the texture component of the image that, to our best knowledge, has never been used before in segmentation schemes, and iii) the curvature component κ is not present in our scheme since it was experimentally determined that it does not provide any significant improvement to the overall segmentation.

Acknowledgments

This research work was supported by the project PYTHAGORAS (of EPEAEK II) which is co-funded by the European Social Fund (75%) and the Greek National Resources (25%). The authors also wish to thank G. Evangelopoulos for his valuable help in the texture analysis part and G. Papandreou and S. Lefkimmiatis for help with pdfLaTeX.

References

[1] S. Beucher and F. Meyer, *The Morphological Approach to Segmentation: The Watershed Transformation*, Mathematical morphology in image processing, 1993.

[2] T. Brox, M. Rousson, R. Deriche, and J. Weickert, *Unsupervised Segmentation Incorporating Colour, Texture, and Motion*, Computer Analysis of Images and Patterns, 2003, pp. 353–360.

[3] V. Caselles, R. Kimmel, and G.Sapiro, *Geodesic Active Contours*, Int'l J. Comp. Vision **22** (1997), no. 1, 61–79.

[4] J. P. Havlicek, D. S. Harding, and A. C. Bovik, *The Multi-component AM-FM Image Representation*, IEEE Trans. Image Processing **5** (1996), no. 6, 1094–1100.

[5] I. Kokkinos, G. Evangelopoulos, and P. Maragos, *Advances in Texture Analysis: Energy Dominant Components and Multiple Hypothesis Testing*, Proc. ICIP, 2004.

[6] R. Malladi, J. A. Sethian, and B. C. Vemuri, *Shape modeling with front propagation: A level set approach*, IEEE Tr. Pattern Anal. Mach. Intel. **17** (1995), no. 2, 158–175.

[7] P. Maragos and A. C. Bovik, *Image Demodulation Using Multidimensional Energy Separation*, J. Opt. Soc. Amer. A **12** (1995), no. 9, 1867–1876.

[8] P. Maragos and M. A. Butt, *Curve evolution, differential morphology, and distance transforms applied to multiscale and eikonal problems*, Fundamenta Informaticae **41** (2000), 91 –129.

[9] D. Martin, C. Fowlkes, and J. Malik, *Learning to Detect Natural Image Boundaries Using Local Brightness, Color, and Texture Cues*, IEEE Tr. Pattern Anal. Mach. Intel. **26** (2004), no. 5, 530 –549.

[10] F. Meyer and P. Maragos, *Multiscale morphological segmentations based on watershed, flooding, and eikonal pde*, Proceedings of scale-space, 1999, pp. 351–362.

[11] ———, *Nonlinear scale-space representation with morphological levelings*, J. Visual Communications. & Image Representation **11** (2000), 245–265.

[12] F. Meyer, *The Levelings*, Proc. fourth international symposium on mathematical morphology and its applications to image processing, 1998June, pp. 199 –206.

[13] Y. Meyer, *Oscillating Patterns in Image Processing and Nonlinear Evolution Equations*, University Lecture Series, vol. 22, AMS, 2002.

[14] L. Najman and M. Schmitt, *Watershed of a continuous function*, Signal Processing **38** (1994), no. 7, 99–112.

[15] S. Osher and J. Sethian, *Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations*, J. Comp. Phys. **79** (1988), 12–49.

[16] J. Rivest and P. Cavanagh, *Localizing contours defined by more than one attribute*, Vision Research **36** (1996), no. 1, 53 –66.

[17] P. Salembier and J. Serra, *Flat zones filtering, connected operators, and filters by reconstruction*, IEEE Trans. Image Processing **4** (1995), no. 8, 153–1160.

[18] J. A. Sethian, *Level set methods and fast marching methods*, Cambridge University Press, 1999.

[19] A. Sofou and P. Maragos, *PDE-based Modeling of Image Segmentation using Volumic Flooding*, Proc. Int'l Conf. Image Processing, 2003.

[20] J. N. Tsitsiklis, *Efficient algorithms for globally optimized trajectories*, IEEE Trans. Automat. Contr. **40** (1995), no. 9, 1528–1538.

[21] L. A. Vese and S. J. Osher, *Modeling textures with total variation minimization and oscillating patterns in image processing*, J. Sci. Comp. **19** (2003), no. 1-3, 553–572.

[22] L. Vincent and P. Soille, *Watershed in digital spaces: An efficient algorithm based on immersion simulations*, IEEE Tr. Pat. Anal. Mach. Intel. **13** (1991June), 583–598.

[23] Y. J. Zhang, *A survey on evaluation methods for image segmentation*, Pattern Recognition **29** (1996), no. 8, 1335–1346.

VIII

ALGORITHMS AND ARCHITECTURES

A partitioned algorithm for the image foresting transform

FELIPE P. G. BERGO and ALEXANDRE X. FALCÃO

Laboratório de Informática Visual (LIV), Instituto de Computação (IC), Universidade Estadual de Campinas (Unicamp), SP, Brazil
`bergo@liv.ic.unicamp.br, afalcao@ic.unicamp.br`

Abstract The Image Foresting Transform (IFT) is a powerful graph-based framework for the design and implementation of image processing operators. In this work we present the Partitioned IFT (PIFT), an algorithm that computes any IFT operator as a series of independent IFT-like computations. The PIFT makes parallelization of existing IFT operators easy, and allows the computation of IFTs in systems with scarce memory. We evaluate the PIFT for two image processing applications: watershed segmentation and Euclidean distance transforms.

Keywords: graph algorithms, parallel algorithms, image foresting transform, distance transforms.

1. Introduction

The Image Foresting Transform (IFT) [9] is a graph-based framework for the design and implementation of image processing operators. It reduces image processing operations, such as watersheds [3, 15], morphological reconstructions [8], skeletonization [7] and distance transforms [5], to the computation of a minimum-cost path forest over an implicit graph representation of the image. The IFT runs in linear time, but it does not take advantage of parallel and distributed computer systems. Its data structures also require considerable memory space [10], and this can be a limitation to the processing of large 3D images.

In this work we present the Partitioned IFT, an algorithm that computes any IFT as a set of independent IFTs over partitions of the input image. Both time and memory required to compute the IFT of each partition are proportional to the size of that partition. The minimum-cost path forests of the partitions are merged by fast differential IFTs [6]. This scheme provides the means to take advantage of parallel and distributed computer systems (by assigning each partition's IFT to a different central processing unit (CPU)) and to allow the computation of IFTs with a reduced memory footprint (by computing partition forests sequentially).

G. Banon, et al. (eds),
Mathematical Morphology and its Applications to Signal and Image Processing, 425–436.
©2007 *MCT/INPE. Printed in Brazil.*
URL of this article: http://urlib.net/dpi.inpe.br/ismm@80/2007/04.13.17.56

2. Related works

2.1 Related algorithms

Moga et al. [12] presented two parallel watershed algorithms that treat the image as a graph and perform independent flooding simulations in image partitions. Parallel flooding simulations are repeated while plateaus overflow to adjacent partitions. The same group [13] presented a similar parallel algorithm for the computation of the watershed-from-markers transform. Both works achieve scalable speedups in parallel architectures, but the speedup factor does not scale linearly with the number of processors. Moga et al. [12] achieve speedup factors[1] around 2 for 4-CPU systems, and 3.5 for 8-CPU systems. Bruno and Costa [4] present a distributed algorithm for the computation of Euclidean distance transforms (EDT) based on morphological dilations. Their algorithm achieves a speedup factor of 3.5 on a 4-CPU system.

2.2 The image foresting transform

The IFT algorithm is essentially Dijkstra's algorithm [1], modified for multiple sources and general path cost functions [9]. The image is interpreted as a directed graph whose nodes are the pixels. The edges are defined implicitly by an *adjacency relation* $\mathcal{A}$. Tree roots are drawn from a set $\mathcal{S}$ of *seed nodes* and path costs are given by a *path cost function* f. We use $P^*(s)$ to denote the current path reaching pixel s, $\langle s \rangle$ to denote a *trivial path* containing a single node, and $\langle s, t \rangle$ to denote the edge from pixel s to pixel t. $P^*(s) \cdot \langle s, t \rangle$ is the path that results from the concatenation of $P^*(s)$ and an edge $\langle s, t \rangle$.

The choice of $\mathcal{A}$, $\mathcal{S}$ and f define an IFT operator. The IFT algorithm can compute by ordered propagation any forest property that uses the seed set as reference. Usually, the IFT computes 4 maps: the cost map C stores the cost of the optimal path that reaches each pixel, the predecessor map P stores the predecessor of each pixel in the forest, the root map R stores the root of each pixel's optimal path, and the label map L stores object labels for each pixel. Algorithm 3 below computes the IFT.

Algorithm 3. IFT.

```
INPUT:      Image $\mathbf{I}$, Path-cost function $f$, Adjacency relation $\mathcal{A}$, Seed set $\mathcal{S}$
            and Seed label map $L_0$.
OUTPUT:     Cost map $C$, Predecessor map $P$, Root map $R$ and Label map $L$.
AUXILIARY:  Priority queue $Q$.

1.  Set $Q \leftarrow \emptyset$.
```

[1]The speedup factor of a parallel algorithm on an n-CPU parallel system is calculated as $\frac{t_1}{t_N}$, where t_1 is the time required to perform the computation on a single-CPU system, and t_N is the time required to perform the computation on an n-way system.

```
2.  For each pixel $s \in \mathbf{I} \setminus \mathcal{S}$, do
3.    Set $C(s) \leftarrow \infty$, $P(s) \leftarrow nil$, $R(s) \leftarrow s$ and $L(s) \leftarrow nil$.
4.  For each pixel $s \in \mathcal{S}$, do
5.    Set $C(s) \leftarrow f(\langle s \rangle)$ and $L(s) \leftarrow L_0(s)$.
6.    Insert $s$ in $Q$.
7.  While $Q \neq \emptyset$, do
8.    Remove a pixel $s$ from $Q$ such that $C(s)$ is minimum.
9.    For each $t$ such that $(s,t) \in \mathcal{A}$, do
10.     Compute $cost \leftarrow f(P^*(s) \cdot \langle s,t \rangle)$.
11.     If $cost < C(t)$ then
12.       If $t \in Q$ then remove $t$ from $Q$.
13.       Set $P(t) \leftarrow s$, $C(t) \leftarrow cost$, $L(t) \leftarrow L(s)$, $R(t) \leftarrow R(s)$.
14.       Insert $t$ in $Q$.
```

Lines 1–3 set the forest to an initial state where every node's optimum path is a trivial path with infinite cost. Lines 4–6 insert the seed pixels in the priority queue with a trivial path cost computed by f, and initialize seed labels for ordered propagation. The loop of Lines 7–14 uses the priority queue to propagate the optimum paths and conquer the entire image. As long as f is finite and smooth [9], an optimum path with finite cost will be assigned to all pixels connected to $\mathcal{S}$. Once a pixel is removed from the queue (Line 8), it is never inserted again. Therefore, the main loop is repeated $|\mathbf{I}|$ times. For integer path costs with limited increments, Q can be efficiently implemented such that insertions and removals take $O(1)$ time [1]. With small adjacency relations ($|\mathcal{A}| \ll |\mathbf{I}|$) and $O(1)$ queue operations, the IFT algorithm runs in $O(|\mathbf{I}|)$ time [9].

Two common path cost functions for IFT operators are f_{max} and f_{euc}, shown in Equations 1–2 below. Both f_{max} and f_{euc} are *smooth*, as required to ensure the correctness of the IFT [9].

$$f_{max}(\langle s_1, \ldots, s_n \rangle) = \begin{cases} max_{i=1}^{n} \left(I \left(s_i \right) \right) & \text{if } n > 1, \\ h(s_1) & \text{otherwise.} \end{cases} \tag{1}$$

$$f_{euc}(\langle s_1, \ldots, s_n \rangle) = \text{Euclidean distance between } s_1 \text{ and } s_n \tag{2}$$

where $I(s)$ is some value associated to pixel s (such as intensity or gradient intensity) and h is a handicap function for trivial paths. A watershed-from-markers transform can be implemented as an IFT where f is f_{max} (Equation 1), $h = 0$ (for marker imposition), $\mathcal{A}$ is an adjacency with radius between 1 and $\sqrt{2}$ and $\mathcal{S}$ contains the watershed markers [6, 9]. A classical watershed can be implemented using $f = f_{max}$, $h(s) = I(s) + 1$ and $\mathcal{S} = \mathbf{I}$ [11]. Function f_{euc} (Equation 2) allows the computation of distance transforms [5], discrete Voronoi diagrams, skeletonizations and shape saliences [2, 7, 9, 14].

2.3 The differential image foresting transform

The differential IFT [6] (DIFT) was motivated by interactive 3D image segmentation applications where the user interactively selects the seed pixels. It is quite common for the user to add new seeds and remove previous ones based on the visualization of the segmentation result. The first IFT is computed by Algorithm 3 as usual, from a seed set $\mathcal{S}_0$. The maps C, P, R and L must be initialized to a forest of trivial paths with infinite costs before the first DIFT is computed. Given a set $\mathcal{S}'$ of seeds to be added and a set $\mathcal{S}''$ of tree roots to be removed, the DIFT computes the optimum path forest for the effective seed seet $\mathcal{S}_1 = (\mathcal{S}_0 \setminus \mathcal{S}'') \cup \mathcal{S}'$. The DIFT processes only pixels affected by the seed set editing, and runs in sublinear time. Instead of providing $\mathcal{S}''$ directly, the DIFT takes a set $\mathcal{M}$ of removal markers, and $\mathcal{S}''$ is computed as the set of roots of the pixels in $\mathcal{M}$. Algorithm 4 below is the main DIFT algorithm. The DIFT-TreeRemoval subroutine referenced in Line 2 visits all pixels that belong to removed trees, sets their optimum paths to trivial paths with infinite costs (forcing their recalculation by Algorithm 4), and builds the set $\mathcal{F}$ of frontier pixels.

Algorithm 4. DIFT.

```
INPUT:      Image I, Cost map C, Predecessor map P, Root map R, Label
            map L, Path-cost function f, Adjacency relation A, Set S' of
            new seed pixels, Set M of marking pixels, Seed label map L_0.
OUTPUT:     C, P, R and L.
AUXILIARY:  Priority queue Q, Frontier set F.

1.  Set $Q \leftarrow \emptyset$.
2.  $(C, P, \mathcal{F}) \leftarrow$DIFT-TREEREMOVAL$(C, P, R, L, \mathcal{A}, \mathcal{M})$.
3.  $\mathcal{F} \leftarrow \mathcal{F} \setminus \mathcal{S}'$.
4.  While $\mathcal{S}' \neq \emptyset$, do
5.      Remove any $t$ from $\mathcal{S}'$.
6.      If $f(\langle t \rangle) < C(t)$ then
7.          Set $C(t) \leftarrow f(\langle t \rangle)$, $R(t) \leftarrow t$, $L(t) \leftarrow L_0(t)$, $P(t) \leftarrow nil$.
8.          Set $\mathcal{F} \leftarrow \mathcal{F} \cup \{t\}$.
9.  While $\mathcal{F} \neq \emptyset$, do
10.     Remove any $t$ from $\mathcal{F}$ and insert $t$ in $Q$.
11. While $Q \neq \emptyset$, do
12.     Remove a pixel $s$ from $Q$, such that $C(s)$ is minimum.
13.     For each $t$ such that $(s, t) \in \mathcal{A}$, do
14.         Compute $cost \leftarrow f(P^*(s) \cdot \langle s, t \rangle)$.
15.         If $cost < C(t)$ or $P(t) = s$ then
16.             If $t \in Q$ then remove $t$ from $Q$.
17.             Set $P(t) \leftarrow s$, $C(t) \leftarrow cost$, $R(t) \leftarrow R(s)$, $L(t) \leftarrow L(s)$.
18.             Insert $t$ in $Q$.
```

Lines 2–3 compute a set $\mathcal{F}$ of frontier pixels that belong to non-removed trees but share edges with pixels in removed trees. Lines 4–10 insert the new seeds and the frontier pixels in the queue. Lines 11–18 are very much like the main loop of the IFT Algorithm (Algorithm 3), except for the condition $P(t) = s$ in Line 14, which forces the update of all pixels that had their optimum paths modified. The result of the DIFT is an optimum path forest for the "effective seed set" $\mathcal{S}_1 = (\mathcal{S}_0 \setminus \mathcal{S}'') \cup \mathcal{S}'$.

3. The partitioned image foresting transform

In the Partitioned IFT (PIFT), we split the input image and seed set in N_P *partitions*. The number of partitions can be chosen to match the number of available processing nodes, or so that the computer system has enough memory to run the IFT algorithm on each image partition. Partitions do not need to be equally sized. We compute independent IFTs on each partition. At this point, we have an optimum forest that ignores the inter-partition edges of the graph. Figure 1(a) shows an example of this partial result for the EDT using a set of random pixels as seeds and 3 partitions. To allow propagation through the inter-partition graph edges, we consider the paths obtained by the concatenation of each edge $\langle s,t\rangle$ to $P^*(s)$ (Figure 1(c)). When $f(P^*(s)\cdot\langle s,t\rangle)$ is less than the current cost of t, or the edge was part of the destination pixel's previous optimal path, the endpoint is added as seed in a differential IFT so that it can be propagated. If more than one inter-partition edge share a same endpoint t, the one that provides the lower path cost $P^*(t)$ is propagated. A new iteration of differential IFTs is computed for each partition. The PIFT halts when no inter-partition edge satisfies the criteria for addition. Figure 1(b) shows the complete EDT, obtained after 2 iterations over the 3 partitions.

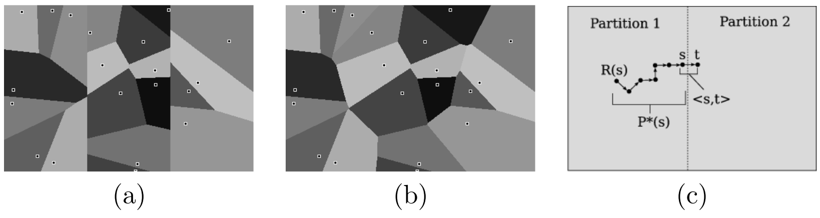


Figure 1. Labels of an EDT with the Partitioned IFT: (a) Partial result after the first iteration and (b) final result after the second iteration. (c) PIFT notation: $\langle s,t\rangle$ is an inter-partition edge, $P^*(s)$ is the optimum path assigned to s, and $R(s)$ the root of $P^*(s)$.

The differential IFTs used in the Partitioned IFT always have an empty set of removal markers. The Partition-IFT algorithm below (Algorithm 5) computes the IFT within a partition. It is essentially the differential IFT algorithm without tree removal, and with special treatment of inter-partition edges.

Algorithm 5. PARTITION-IFT.

INPUT:	*Image partition $\mathbf{I}'$, Cost map C, Predecessor map P, Root map R, Label map L, Path-cost function f, Adjacency relation $\mathcal{A}$, Set $\mathcal{S}$ of seed pixels, Seed label map L_0, Set $\mathcal{E}_I$ of incoming inter-partition edges.*
OUTPUT:	*Maps C, P, R, L and Set $\mathcal{E}_O$ of outgoing inter-partition edges.*
AUXILIARY:	*Priority queue Q.*

```
1.  Set $Q \leftarrow \emptyset$, $\mathcal{E}_O \leftarrow \emptyset$.
2.  If $\mathcal{S} \neq \emptyset$ then
3.      For each pixel $s \in \mathbf{I}'$, do
4.          Set $C(s) \leftarrow \infty$, $P(s) \leftarrow nil$, $R(s) \leftarrow s$ and $L(s) \leftarrow nil$.
5.      For each pixel $s \in \mathcal{S}$, do
6.          Set $C(s) \leftarrow f(\langle s \rangle)$ and $L(s) \leftarrow L_0(s)$.
7.          Insert $s$ in $Q$.
8.  For each edge $\langle s,t \rangle \in \mathcal{E}_I$, do
9.      Compute $cost \leftarrow f(P^*(s) \cdot \langle s,t \rangle)$.
10.     If $cost < C(t)$ or $P(t) = s$ then
11.         Set $C(t) \leftarrow cost$, $P(t) \leftarrow s$, $R(t) \leftarrow R(s)$ and $L(t) \leftarrow L(s)$.
12.         Insert $t$ in $Q$.
13. While $Q \neq \emptyset$, do
14.     Remove a pixel $s$ from $Q$, such that $C(s)$ is minimum.
15.     For each $t$ such that $(s,t) \in \mathcal{A}$, do
16.         If $t \in \mathbf{I}'$ then
17.             Compute $cost \leftarrow f(P^*(s) \cdot \langle s,t \rangle)$.
18.             If $cost < C(t)$ or $P(t) = s$ then
19.                 If $t \in Q$ then remove $t$ from $Q$.
20.                 Set $P(t) \leftarrow s$, $C(t) \leftarrow cost$, $R(t) \leftarrow R(s)$, $L(t) \leftarrow L(s)$.
21.                 Insert $t$ in $Q$.
22.         Else Insert $\langle s,t \rangle$ in $\mathcal{E}_O$.
```

The DIFT is unable to tell whether the algorithm is on the first iteration, therefore the initial state of the forest must be set before the first iteration. In the PIFT, the seed set $\mathcal{S}$ will only be non-empty in the first iteration. We use this property to initialize the partition's forest to trivial paths with infinite costs in Lines 2–4. Lines 5–7 queue and initialize the seed pixels in the same way the IFT does. Lines 8–12 process the incoming inter-partition edges $\mathcal{E}_I$. Edges that offer lower costs to their endpoints or belonged to the previous forest are queued for propagation. If multiple edges in $\mathcal{E}_I$ reach the same endpoint, the edge that provides the lower cost for the endpoint takes precedence. The main loop in Lines 13–22 is very similar to the main loop of the DIFT, with the addition of the partition test $t \in \mathbf{I}'$ in Line 16. Edges within the current partition are processed normally. Inter-partition edges are added to the outgoing edge set $\mathcal{E}_O$ (Line 22). Note that the cost computation in Line 9 may require additional information about $P^*(s)$, which can contain pixels of several partitions. All path information required to compute $f(P^*(s) \cdot \langle s,t \rangle)$ must be passed along with the set $\mathcal{E}_I$.

For f_{max}, only $C(s)$ is required. For f_{euc}, only $R(s)$ is required. Since $L(s)$ may be propagated in Line 11, it must also be part of the input. Passing each element of $\mathcal{E}_I$ as $\{s, t, C(s), R(s), L(s)\}$ is enough to compute the PIFT with either f_{max} or f_{euc}. The PIFT algorithm (Algorithm 6) that computes the IFT of an image $\mathbf{I}$ from its partitions is shown below.

Algorithm 6. PARTITIONED IFT.

```
INPUT:      Image $\mathbf{I}$, Path-cost function $f$, Adjacency relation $\mathcal{A}$, Set $\mathcal{S}$ of
            seed pixels, Seed label map $L_0$, Number of partitions $N_P$.
OUTPUT:     Cost map $C$, Predecessor map $P$, Root map $R$, Label map $L$.
AUXILIARY:  Edge sets $\mathcal{E}$, $\mathcal{E}'$, $\mathcal{E}''$ and $\mathcal{E}'''$, Seed set $\mathcal{S}'$.

1.  Set $\mathcal{E} \leftarrow \emptyset$.
2.  Split $\mathbf{I}$ in $N_P$ partitions $\mathbf{I}[1] \ldots \mathbf{I}[N_P]$.
3.  For $i = 1$ to $N_P$, do
4.      Set $\mathcal{S}' = \{s \mid s \in \mathcal{S} \wedge s \in \mathbf{I}[i]\}$.
5.      Set $(C[i], P[i], R[i], L[i], \mathcal{E}') \leftarrow$
          PARTITION-IFT$(\mathbf{I}[i], C[i], P[i], R[i], L[i], f, \mathcal{A}, \mathcal{S}', L_0, \emptyset)$.
6.      Set $\mathcal{E} \leftarrow \mathcal{E} \cup \mathcal{E}'$.
7.  Repeat
8.      Set $\mathcal{E}''' \leftarrow \emptyset$.
9.      For $i = 1$ to $N_P$, do
10.         Set $\mathcal{E}'' = \{\langle s,t \rangle \mid \langle s,t \rangle \in \mathcal{E} \wedge t \in \mathbf{I}[i]\}$.
11.         Set $(C[i], P[i], R[i], L[i], \mathcal{E}') \leftarrow$
              PARTITION-IFT$(\mathbf{I}[i], C[i], P[i], R[i], L[i], f, \mathcal{A}, \emptyset, nil, \mathcal{E}'')$.
12.         Set $\mathcal{E}''' \leftarrow \mathcal{E}''' \cup \mathcal{E}'$.
13.     Set $\mathcal{E} \leftarrow \mathcal{E}'''$.
14.     Until $\mathcal{E} = \emptyset$.
15. Set $C \leftarrow \cup_{i=1}^{N_P} C[i]$, $P \leftarrow \cup_{i=1}^{N_P} P[i]$, $R \leftarrow \cup_{i=1}^{N_P} R[i]$ and $L \leftarrow \cup_{i=1}^{N_P} L[i]$.
```

Lines 1–2 initialize the inter-partition edge set $\mathcal{E}$ and split the input image in N_P partitions. The loop in Lines 3–6 run the first IFT iteration on each partition. All inter-partition edges are accumulated in the set $\mathcal{E}$. The loop in Lines 7–14 run the remaining IFT iterations on the partitions, until no propagation occurs and the set $\mathcal{E}$ of inter-partition edges is empty (Line 14).

For parallel architectures, both loops (Lines 3–6 and 7–14) can be done in parallel. For distributed systems, the executions of Partition-IFT (Algorithm 5) can be performed as remote procedure calls. Note that the partitioned maps ($C[i]$, $P[i]$, $R[i]$ and $L[i]$) are only needed at the end of the algorithm, to compose the final IFT maps. In a distributed implementation, these maps can be kept on the remote processing nodes and do not need to be transferred at each call to Partition-IFT, as they are not modified by the caller.

Performance Considerations. The overall number of pixels processed by the PIFT is larger than $|\mathbf{I}|$. After the loop of Lines 3–6 of Algorithm 6, the PIFT has already processed $|\mathbf{I}|$ nodes. However, the number of pixels processed by the loop of Lines 7–14 decreases at each iteration, and the algorithm converges rapidly to the optimum path forest. The number of PIFT iterations — i.e., one iteration of the loop of Lines 3–6 plus the number of iterations of the loop of Lines 7–14 — is bounded by the maximum number of inter-partition edges contained by an optimum path, plus one. Each inter-partition edge postpones the resolution of the optimum path to the next PIFT iteration. Figure 2 illustrates some examples. In an Euclidean distance transform (Figure 2(a)), all paths flow away from the roots, and a path may cross at most $N_P - 1$ partition boundaries, requiring at most N_P PIFT iterations. For path cost functions like f_{max}, there is no restriction to the shape of optimum paths, and cases like the one in Figure 2(b) can occur. However, as the number of iterations increases, the number of pixels processed by each iteration tends to decrease, and the PIFT converges more rapidly to the optimum forest.

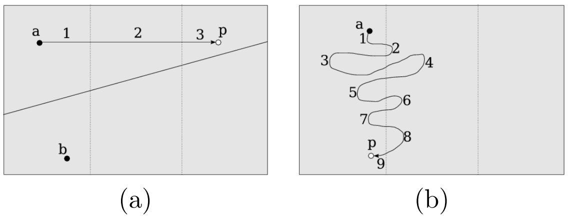


Figure 2. Partition crossings and PIFT iterations: In the PIFT-EDT, paths cross at most $N_P - 1$ partition boundaries. In (a), $P^*(p)$ crosses 2 boundaries to reach p from a. The numbers are the iteration in which the path segment is propagated. (b) For general path-cost functions, a path may cross partition boundaries several times.

4. Experimental results

We implemented the PIFT as a client-server system, with a simple TCP stream-based protocol for communication between the master client that executes Algorithm 6 and the distributed servers that execute Algorithm 5. In our implementation, the image is always split in equal-sized partitions, using the x coordinate to separate partitions (such as in Figure 1(a)). We chose 3 applications to evaluate the PIFT:

1. WS-BRAIN: Watershed-based segmentation of a 3D MR image of the brain, using $f = f_{max}$ and $\mathcal{A}$ =6-neighborhood adjacency. Seeds were selected interactively in the background and in the brain. The gradient intensity was computed by a Gaussian enhacement filter followed by morphological gradient computation [6]. The size of the image is $356 \times 356 \times 241$, with a voxel size of $0.70mm^3$ (Figure 3(a–c)).

2. EDT-RND: Euclidean distance transform of 1000 random points within a 256^3 volume, using $f = f_{euc}$ and $\mathcal{A}$ =26-neighborhood (Figure 3(d)).

3. EDT-BRAIN: Euclidean distance transform using the border of the brain object (segmented in the first application) as seed set ($|\mathcal{S}| = 355,556$). $f = f_{euc}$, $\mathcal{A}$ =26-neighborhood and volume size is $356 \times 356 \times 241$ (Figure 3(e)).

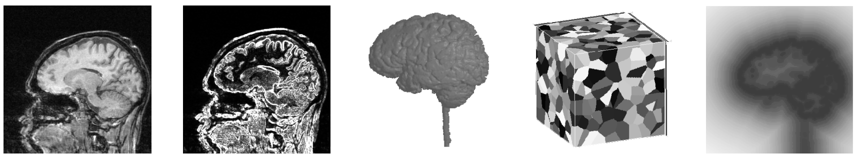


Figure 3. Images from the evaluation applications: (a) Slice from the WS-BRAIN input image. (b) gradient intensity of (a). (c) 3D renderization of the WS-BRAIN result. (d) Visualization of the discrete Voronoi diagram, result of the EDT-RND. (e) Slice from the distance map computed in EDT-BRAIN.

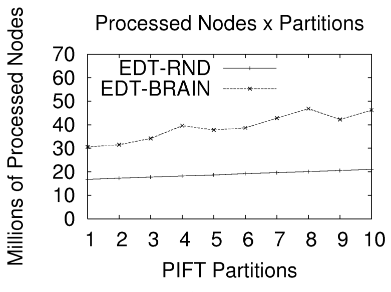
First, we measured the processing overhead of the PIFT as the number of partitions (N_P) increases. We computed the 3 applications with the PIFT, using from 1 to 10 partitions. Table 1 and Figure 4 present the number of nodes processed in each case and the upper bound for the speedup factor. These results indicate that a 10-way parallel system may be able to offer a speedup factor of 6.60 to the EDT computation, and a factor of 2.34 to the Watershed transform on these instances of problems.

The EDT computations required at most 4 iterations before halting. PIFTs based on f_{max} are less efficient, since they allow free-form paths that can traverse several partitions. This can be noticed by the irregularity and increased slope of the plot in Figure 4(b), as compared to Figure 4(a). The WS-BRAIN PIFTs required at most 23 iterations to converge. The number of processed nodes grows linearly with the number of partitions. In real data with non-uniform distributions (WS-BRAIN and EDT-BRAIN), bad choices of partition boundaries may increase the number of processed nodes, such as in the $N_P = 4$ and $N_P = 8$ cases of EDT-BRAIN and $N_P = 5$ of WS-BRAIN.

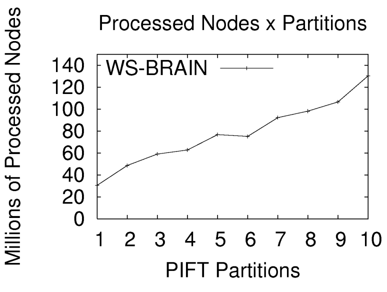
In a second set of experiments we used the PIFT to compute EDT-RND, EDT-BRAIN and WS-BRAIN in two parallel systems: a PC with 2 CPUs (Athlon MP 1800+@1150 MHz) and 2 GB of RAM, and a Compaq AlphaServer GS140 6/525 with 10 CPUs (Alpha EV6@525 MHz) and 8 GB of RAM. Table 2 presents the results. On the EDT applications, we achieved speedup factors very close to the measured upper bounds (Table 1) for $N_P = 2$ and $N_P = 4$. On other hand, there was little or no speedup for

Table 1. Number of processed nodes and upper bound for the speedup factor in each application, using up to 10 partitions.

N_P	**WS-BRAIN**		**EDT-RND**		**EDT-BRAIN**	
	Nodes	Speedup	Nodes	Speedup	Nodes	Speedup
1	30.5×10^6	1.00	16.8×10^6	1.00	30.5×10^6	1.00
2	48.6×10^6	1.25	17.3×10^6	1.94	31.5×10^6	1.93
3	59.0×10^6	1.55	17.7×10^6	2.84	34.2×10^6	2.67
4	62.7×10^6	1.94	18.2×10^6	3.69	39.6×10^6	3.08
5	76.7×10^6	1.98	18.6×10^6	4.51	37.8×10^6	4.03
6	75.1×10^6	2.43	19.2×10^6	5.25	38.7×10^6	4.72
7	92.2×10^6	2.31	19.7×10^6	5.96	42.8×10^6	4.98
8	98.1×10^6	2.48	20.1×10^6	6.68	46.8×10^6	5.21
9	106.5×10^6	2.57	20.5×10^6	7.37	42.2×10^6	6.50
10	130.2×10^6	2.34	21.0×10^6	8.00	46.2×10^6	6.60



(a)



(b)

Figure 4. Number of processed nodes vs. number of partitions for (a) EDT-RND, EDT-BRAIN and (b) WS-BRAIN.

the watershed application. Our prototype implementation uses a naive communication protocol with no data compression. Besides that, the edge set transfers of Lines 5 and 11 of Algorithm 6 were implemented in a sequential way, and instances with a large number of partitions and/or a large number of PIFT iterations (such as WS-BRAIN with $N_P = 10$) performed poorly because the CPUs remained idle while waiting for the client to complete the sequential edge set transfers.

Table 2. PIFT performance on two parallel computer systems. Times are given in seconds.

System	N_P	**WS-BRAIN**		**EDT-RND**		**EDT-BRAIN**	
		Time	Speedup	Time	Speedup	Time	Speedup
Dual Athlon	1	258.1	1.00	195.9	1.00	459.5	1.00
	2	242.9	1.06	106.2	1.84	246.3	1.87
10-CPU GS140	1	280.6	1.00	228.8	1.00	611.4	1.00
	2	284.3	0.99	126.6	1.81	324.2	1.89
	4	226.2	1.24	73.0	3.13	274.3	2.23
	8	249.3	1.13	49.3	4.64	214.1	2.86
	10	336.4	0.83	47.9	4.78	197.7	3.09

5. Conclusion and future works

We introduced the Partitioned Image Foresting Transform, an algorithm that computes minimum-cost path forests as a set of independent DIFTs [6, 9] in partitions of the input image. The PIFT is useful for taking advantage of parallel computer systems and for computing IFTs in computer systems with limited memory, such as handhelds and embedded systems. The PIFT is applicable to any IFT-based operator, and therefore can be readily employed to parallelize morphological reconstructions [8], watershed transforms [2, 3, 6, 15], distance transforms [5, 9] and skeletonizations [7, 14], among other operators. It is a trend in microprocessor technology to compensate CPU speed limitations by producing multi-core CPUs. The PIFT is an important contribution that allows existing image processing applications to use modern hardware efficiently with minimum effort.

We implemented a prototype PIFT system with a simple client-server architecture built on top of TCP streams. Even with no data compression and with some inneficient network operations, we achieved speedup factors very close to the expected upper bounds for EDT operations. PIFT-based watershed segmentation performed poorly due to the inneficiency of edge set transfers in our prototype. With a better protocol, the PIFT should be able to reach speedup factors closer to the upper bounds in Table 1.

Future works include: development of better protocols for implementation of the PIFT in parallel systems, evaluation of the speedup bounds for specific operators – such as the watershed transform – and investigation of enhancements to the PIFT such as partitioning schemes and iteration scheduling among nodes.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms and Applications*, Prentice-Hall, 1993.

[2] F. P. G. Bergo and A. X. Falcão, *Fast and automatic curvilinear reformatting of MR images of the brain for diagnosis of dysplastic lesions*, Proc. of 3rd Intl. Symp. on Biomedical Imaging (April 2006), 486–489.

[3] S. Beucher and F. Meyer, *The morphological approach to segmentation: The watershed transformation*, Marcel Dekker, 1993.

[4] O. M. Bruno and L. F. Costa, *A parallel implementation of exact Euclidean distance transform based on exact dilations*, Microprocessors and Microsystems **28** (April 2004), no. 3, 107–113.

[5] P. E. Danielsson, *Euclidean Distance Mapping*, Computer Graphics and Image Processing **14** (1980), 227–248.

[6] A. X. Falcão and F. P. G. Bergo, *Interactive Volume Segmentation with Differential Image Foresting Transforms*, IEEE Trans. on Medical Imaging **23** (2004), no. 9, 1100–1108.

[7] A. X. Falcão, L. F. Costa, and B. S. da Cunha, *Multiscale skeletons by image foresting transform and its applications to neuromorphometry*, Pattern Recognition **35** (April 2002), no. 7, 1571–1582.

[8] A. X. Falcão, B. S. da Cunha, and R. A. Lotufo, *Design of connected operators using the image foresting transform*, Proc. of SPIE on Medical Imaging **4322** (February 2001), 468–479.

[9] A. X. Falcão, J. Stolfi, and R. A. Lotufo, *The Image Foresting Transform: Theory, Algorithms, and Applications*, IEEE Trans. on Pattern Analysis and Machine Intelligence **26** (2004), no. 1, 19–29.

[10] P. Felkel, M. Bruckschwaiger, and R. Wegenkittl, *Implementation and Complexity of the Watershed-from-Markers Algorithm Computed as a Minimal Cost Forest*, Computer Graphics Forum (Eurographics) **20** (2001), no. 3, C26–C35.

[11] R. A. Lotufo and A. X. Falcão, *The ordered queue and the optimality of the watershed approaches*, Mathematical Morphology and its Applications to Image and Signal Processing **18** (June 2000), 341–350.

[12] A. N. Moga, B. Cramariuc, and M. Gabbouj, *Parallel watershed transformation algorithms for image segmentation*, Parallel Computing **24** (December 1998), no. 14, 1981–2001.

[13] A. N. Moga and M. Gabbouj, *Parallel Marker-Based Image Segmentation with Watershed Transformation*, Journal of Parallel and Distributed Computing **51** (May 1998), no. 1, 27–45.

[14] R. S. Torres, A. X. Falcão, and L. F. Costa, *A graph-based approach for multiscale shape analysis*, Pattern Recognition **37** (June 2004), no. 6, 1163–1174.

[15] L. Vincent and P. Soille, *Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations*, IEEE Trans. on Pattern Analysis and Machine Intelligence **13** (June 1991), no. 6.

1D Component tree in linear time and space and its application to gray-level image multithresholding

David Menotti[1, 2], Laurent Najman[1] and Arnaldo de A. Araújo[2]

[1] *Université Paris-Est (UPE), LABINFO-IGM, UMR CNRS 8049, A2SI-ESIEE, France*
`{d.menotti,l.najman}@esiee.fr`

[2] *Universidade Federal de Minas Gerais (UFMG), Belo Horizonte, Brazil*
`{menotti,arnaldo}@dcc.ufmg.br`

Abstract The upper-weighted sets of a signal are the sets of points with weight above a given threshold. The components of the upper-weighted sets, thanks to the inclusion relation, can be organized in a tree structure, which is called the component tree. In this work, we present a linear time and space algorithm to compute the component tree of one-dimensional signals. From this algorithm we derive an efficient gray-level image multithresholding method, which is based on the hypothesis that objects which appear on an image can be represented by salient classes present in the histogram of this image. These classes are modelled as the most significative components of the histogram's component tree. We show results of the proposed method and compare it with classical methods.

Keywords: component-tree, weighted ordered sets, multithresholding.

1. Introduction

The upper-weighted sets of a signal are the sets of points with weight above a given threshold. The components of the upper-weighted sets, thanks to the inclusion relation, can be organized into a tree structure, that is called the component tree. The component tree captures some essential features of a signal. It has been used (under several variations) in numerous applications including image filtering and segmentation [5], video segmentation [13], image registration [9], image compression [13]. In the literature, there are several algorithms to compute the component tree [2, 8, 10, 13]. The algorithm with the best time complexity to compute the component tree for N dimensional signals (e.g., a mapping from $\mathbb{Z}^N$ to $\mathbb{Z}$, where $N \in \mathbb{N}$) was recently proposed in [10] and it is quasi linear.

In this work, we propose a time and space linear algorithm to compute the component tree of a weighted ordered set (WOS), i.e., a model of 1D signals. As a possible application, we propose a gray-level image multithresholding method for image segmentation, which is based on the hypothesis that objects which appear on an image can be represented by salient classes present in the histogram of the image. These salient classes are modelled as the most significative components of the component tree, where the importance corresponds to the volume attribute.

The remaining of this paper is organized as follows. In Section 2, we introduce definitions for WOS and define the component tree in this framework. An algorithm to compute the component tree in linear time and space for WOS is presented in Section 3. In Section 4, we introduce a new method for gray-level image multithresholding. An experimental comparison with classical methods is performed. In Section 5, conclusions are pointed out.

2. Weighted ordered sets and the component tree

In this section we introduce the notion of WOS, which allows us to model 1D signals, and the component tree of such WOS.

2.1 Basic notions for ordered set

Let P be a finite set of points and let $\prec$ be a binary relation on P (i.e., a subset of the Cartesian product $P \times P$) which is transitive ($(x, y) \in \prec, (y, z) \in \prec \Rightarrow (x, z) \in \prec$), and trichotomous (i.e., exactly one of $(x, y) \in \prec, (y, x) \in \prec$ and $x = y$ is true). The relation $\prec$ defines an (total) order on P, and the pair $(P, \prec)$ is a *(totally) ordered set.* Let $(P, \prec)$ be an ordered set and let $x, y, z \in P$. If $(x, y) \in \prec$ and there is no z such that $(x, z) \in \prec$ and $(z, y) \in \prec$, then we say that y is the *successor* of x and x is the *predecessor* of y. Let $(P, \prec)$ be an ordered set. Let $X = \{x_0, x_1, ..., x_n\}$ be a subset of points of P where $x_0, x_1, ..., x_n$ are arranged in increasing order. If for any $i \in [1, n]$, x_i is the successor of x_{i-1}, then we say that X is a *connected set.* We also say that x_0 and x_n are the starting and the ending points of X, respectively.

2.2 Basic notions for weighted ordered set

We denote by $\mathcal{F}(P, D)$, or simply by $\mathcal{F}$, the set composed of all mappings from P to D, where D is any set equipped with a total order e.g., the set of rational numbers or the set of integers). For a mapping $F \in \mathcal{F}$, the triplet $(P, \prec, F)$ is called a *weighted ordered set* (WOS). For a point $p \in P$, $F(p)$ is called the *weight* (or *level*) of p. Let $F \in \mathcal{F}$ and $h \in D$, we define the *h upper-weighted set of* F, denoted by F_h, as $\{p \in P | F(p) \geq h\}$. A connected set X of an upper-weighted set F_h, which is maximal (i.e., $X = Y$ whenever $X \subseteq Y \subseteq P$ and Y is connected), is called a (h-*weighted*) *connected component* (of F). A h-weighted connected component of F that does not contain a $(h + 1)$-weighted connected component of F is called a (*regional*) *maximum* of F. We define $h_{min} = min\{F(p)|p \in P\}$ and

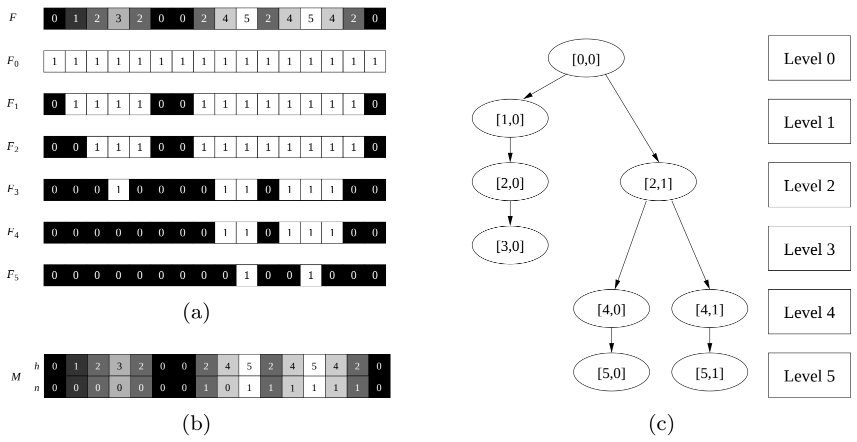


Figure 1. A component tree example. (a) A weighted ordered set $(P, \prec, F)$ and its upper-weighted sets at weights 0, 1, 2, 3, 4 and 5. (b) The associated component mapping M. (c) The component tree $\mathcal{C}(F)$ of F.

$h_{max} = max\{F(p)|p \in P\}$ as the minimum and the maximum weights in the mapping F, respectively.

Figure 1(a) shows a WOS $(P, \prec, F)$ with 16 points and the 6-upper-weighted sets of F, from $h_{min} = 0$ to $h_{max} = 5$. The set F_5 is made of two connected components which are regional maxima of F. The set F_3, in turn, is made of three connected components - one of them being a regional maximum of F.

2.3 Component tree

From the example shown in Figure 1(a) we observe that the weighted connected components of the different upper-weighted sets may be organized to form a tree structure, thanks to the inclusion relation.

Let $F \in \mathcal{F}$ and let $s \subseteq P$ be a connected component of F. We set $f(s) = max\{h|s$ is a (h-weighted) connected component of $F\}$. Note that $f(s) = min\{F(p)|p \in s\}$. Let $h = f(s)$, we say that s is a *(h-weighted) (proper) component of* F. We define $\mathcal{C}(F)$ as the set of all components of F. Let $F \in \mathcal{F}$ and let x and y be distinct elements of $\mathcal{C}(F)$. We say that x is the *parent* of y if $y \subset x$ and there is no other $z \in \mathcal{C}(F)$ such that $y \subset z \subset y$. In this case, we also say that y is the *child* of x. In a parent-children relationship, $\mathcal{C}(F)$ forms a directed tree named *component tree* of F, which will also be denoted by $\mathcal{C}(F)$ by abuse of terminology.

Any element of $\mathcal{C}(F)$ is called a *node*. The node that has no parent, in turn, is called the *root* of the component tree. In the following, for the sake of algorithm description, we denote by $c_{h,n}$ the $(n+1)$ -th h-weighted component of $\mathcal{C}(F)$, where the order of the h-weighted components is derived from the order of their starting point in $(P, \prec)$. In applications, we need to recover the component to which a given point belongs to. For such aim, let us consider the *component mapping* M defined for any point $p \in P$ by $M(p) = [h, n]$, where $h = f(p)$ and $c_{h,n}$ contains p.

Figure 1(c) and Figure 1(b) show the component tree of the WOS depicted in Figure 1(a) and the associated component mapping, respectively. The component $c_{0,0}$ at weight 0 is associated with the node $[0, 0]$, the component $c_{1,0}$ at weight 1 is associated with the node $[1, 0]$, and so on.

3. Linear component tree algorithm for WOS

3.1 Description

In this work, we build the component tree $\mathcal{C}(F)$ of a WOS $(P, \prec, F)$ (i.e., 1D signal) by detecting the components and the parent-children relationship among them. Simultaneously, we build its respective component mapping M. The components of $\mathcal{C}(F)$ are detected in the WOS by analyzing the connected components of F. In an 1D space, the connected components of the upper-weighted sets can be determined by their limits, i.e., the starting and the ending points of the connected components. The connected component limits of upper-weighted sets provide the component limits and, therefore, information for the detection of the components.

In fact, in order to build the component tree we do not need to know exactly the position of the components in the WOS. What we need is to know the components hierarchy, since they respect an inclusion relation. In order to build the component tree in linear time, we propose to analyze the WOS from the starting point up to the ending point. By processing the WOS point by point, we determine every connected components and, consequently, the components present in the WOS with a single scan. In this same scan, we can also establish the hierarchy of the components necessary to create the parent-children relationship. Adopting this approach we can compute the component tree for 1D signals in linear time.

The proposed algorithm roughly works as follows. For each point in the WOS, it checks if the point is a starting point, an ending point, or an inner point (a point which is neither a starting nor an ending point) of a component of F. During this analysis of the WOS points, if a component indicated by a point is found to have descendants it is stored into a stack. The stack plays a fundamental role to maintain the hierarchy of the component tree, as the parent-children relationships are created as edges between parent and child components. In the following paragraphs we explain in details how the component tree and component mapping are build.

The first point, or the starting point p in the WOS $(P, \prec, F)$ receives a special treatment. It belongs to the first component at weight $p_h = F(p)$, and receives the label 0 at the weight p_h. Hence, the node $[p_h, 0]$ is associated with the point p on the component mapping. Once the starting point has been analyzed, we consider the next points. For the sake of simplicity, consider the point p as the current point being analyzed and suppose we want to make decisions about the component, indicated by its predecessor r. We first analyze the weights $p_h = F(p)$ and $r_h = F(r)$, we can find three possibilities: $p_h > r_h$, $p_h = r_h$, and $p_h < r_h$, as shown in Figures 2(a), 2(b), and 2(c).

In the first case, where $p_h > r_h$ (Figure 2(a)), we create a new component at weight p_h, that is, p is the starting point of a component and receives a new label p_n at weight p_h. The node $[p_h, p_n]$ is associated with the point p on the component mapping. Since the node $[r_h, r_n]$ has at least one descendant, i.e., $[p_h, p_n]$, it will be inserted into the stack. Note that no other component with weight smaller or equal to p_h will be inserted into the stack while the component $[p_h, p_n]$ is there.

In the second case, where $p_h = r_h$ (Figure 2(b)), we know that the point p belongs to the same component indicated by the point r. Therefore, the node $[r_h, r_n]$ is associated with the point p, which is the same node as the one which contains r, in the component mapping.

In the last case, where $p_h < r_h$ (Figure 2(c) — an ending or inner point), we know for certain that the component to which r belongs to is already analyzed, i.e., the point r is the ending point of the component. In this situation, we have to decide which component is the parent of the node $[r_h, r_n]$. This decision is based onto the relationship of the nodes in the stack (nodes with descendants to be analyzed) and the node $[p_h, p_n]$. Four scenarios might appear here, as shown in Figures 2(c), 2(d), 2(e), and 2(f).

The first scenario involves the stack being empty. If there are no elements left on the stack (Figure 2(c)), we conclude that the node $[p_h, p_n]$ is the parent of $[r_h, r_n]$. In this situation, we know that the point p belongs to a new component, which is assigned a new label, i.e., p_n, at weight p_h. Hence, the node $[p_h, p_n]$ is associated with the point p on the component mapping. Note that in this case, the point p is not the starting point of this component.

In the other three scenarios assume that there are elements left on the stack, and consider that the stack head element corresponds to the node $[q_h, q_n]$. If $[q_h, q_n]$ has its weight q_h smaller than p_h, i.e., $p_h > q_h$ (Fig-

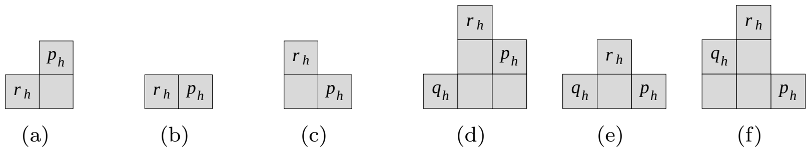


Figure 2. Possible predecessors (r's) of p and the disposition of p in relation to the stack (q's): (a) $p_h > r_h$; (b) $p_h = r_h$; (c) $p_h < r_h$; (d) $p_h > q_h$; (e) $p_h = q_h$; (f) $p_h < q_h$.

ure 2(d)), we have the same situation as when there are no elements on the stack.

In contrast, if $[q_h, q_n]$ has its weight q_h equal to p_h, i.e., $p_h = q_h$ (Figure 2(e)), we observe that the point p belongs to the same component as q, i.e., $p_h = q_h$ and $p_n = q_n$ (the node $[p_h, p_n]$ is associated with the point p on the component mapping). Hence, the node $[p_h, p_n]$ (or $[q_h, q_n]$) is the parent of $[r_h, r_n]$. In this case, the node $[q_h, q_n]$ is removed from the stack, since the possible descendant components between the points q and p have already been computed. Nevertheless, the node $[q_h, q_n]$ can be inserted again into the stack later.

The last possible scenario shows that the node $[q_h, q_n]$ has its weight q_h greater than p_h, i.e., $p_h < q_h$ (Figure 2(f)). Here we have that $[q_h, q_n]$ is the parent of $[r_h, r_n]$. The node $[q_h, q_n]$ is removed from the stack, since $[p_h, p_n]$ has a weight smaller than it. This is necessary to keep the consistency of the stack.

After the node is removed from the stack, we need to find its parent. The node $[p_h, p_n]$ or the new node on the stack head are candidate parents of the removed node $[q_h, q_n]$. The decision about which component is the parent is done by naming the removed component $[q_h, q_n]$ as $[r_h, r_n]$, and starting the decision process again according to the situations presented in Figures 2(c), 2(f), 2(d), and 2(e), as described previously.

Once all points in the WOS were processed, we can still have some components left on the stack. In this case, the component on the stack head is the parent of the component pointed by the ending point of the WOS. The next component on the stack, if there is any, is the parent of the stack head, and so on. These components are removed from stack one by one, and edges are inserted between the parent and child components such that the parent-children relationship is finished. When the stack is empty the component tree is complete.

3.2 Implementation

Algorithm **??** shows an implementation (with low level details) of the algorithm described in Section 3.1 to compute in linear time and space the component tree $\mathcal{C}(F)$, the component mapping M of a WOS $(P, \prec, F)$. The component tree structure (CT) obtained from the algorithm is composed of vectors which store pairs of the child and parent components (the parent-children relationship). Another vector, *nnodes*, composes the CT structure. It is used to indicate the number of nodes at each weight. Thus, the vector *nnodes* is used to generate unique labels for the new nodes at each weight during the processing of the WOS. The stack used, CP (to store pairs $[q_h, q_n]$), implement four basic operations: `StackPush`, `StackPop`, `Stack-Empty`, and `StackView`. In order to build the parent-children relationship of the component tree, the function `InsEdge` is used to insert edges between nodes in the CT structure.

Algorithm 1: BuildComponentTree.

```
Data: $(P, \prec, F)$ - weighted ordered set with $n$ points
Result: $CT$ - component tree structure
Result: $M$ - a map from $P$ to $[h_{min}...h_{max}, 0...n-1]$
$CT.nnodes[F(0)]++$ ; $M(0) \leftarrow [F(0), 0]$;
for $i \leftarrow 1$ ; $i < n$ ; $i++$ do
    $p_h \leftarrow F(i)$ ; $[r_h, r_n] \leftarrow M(i-1)$;
    if $(p_h > r_h)$ then
        $p_n \leftarrow CT.nnodes[p_h]++$ ; $M(i) \leftarrow [p_h, p_n]$;
        StackPush($CP, [r_h, r_n]$);
    else if $(p_h = r_h)$ then
        $p_n \leftarrow r_n$ ; $M(i) \leftarrow [p_h, p_n]$;
    else if $(p_h < r_h)$ then
        while (!StackEmpty($CP$)) do
            $[q_h, q_n] \leftarrow$ StackView($CP$);
            if $(p_h \geq q_h)$ then break;
            InsEdge($CT, [q_h, q_n], [r_h, r_n]$) ;
            StackPop($CP$);
            $[r_h, r_n] \leftarrow [q_h, q_n]$;
        if (StackEmpty($CP$) and $(p_h < r_h)$) or $(p_h > q_h)$ then
            $p_n \leftarrow CT.nnodes[p_h]++$ ; $M(i) \leftarrow [p_h, p_n]$;
            InsEdge($CT, [p_h, p_n], [r_h, r_n]$);
        else if $(p_h = q_h)$ then
            $p_n \leftarrow q_n$; $M(i) \leftarrow [p_h, p_n]$;
            InsEdge($CT, [p_h, p_n], [r_h, r_n]$);
            StackPop($CP$);
while (!StackEmpty($CP$)) do
    $[q_h, q_n] \leftarrow$ StackPop($CP$);
    InsEdge($CT, [q_h, q_n], [p_h, p_n]$);
    $[p_h, p_n] \leftarrow [q_h, q_n]$;
DefineRoot($CT, [p_h, p_n]$);
```

3.3 Complexity analysis

Initially, we stated upper bounds for the data structures used in Algorithm **??** in order to perform the space complexity (SC) analysis. Let n denote the numbers of points in the WOS $(P, \prec, F)$, i.e., $n = |P|$, and let m denote the ordered set amplitude, i.e., $m = h_{max} - h_{min} + 1$. The size of vector *nnodes* corresponds to the domain's size of the mapping F, i.e., the SC of *nnodes* is $O(m)$. A WOS with n points can have a maximum of n components (when each component is composed of a single point). Therefore, the maximum size of the stacks is the number of points in the ordered set, i.e., the SC of CP is $O(n)$. Now we consider the rooted tree. It is a tree with a root node, where every node has a single parent, but the root node does not have a parent. From that we have that all rooted trees with e nodes have $e-1$ edges. As a component tree is a rooted tree, the component tree of a WOS can have at maximum $n-1$ edges. The vectors inside the CT structure have a maximum of n elements ($n-1$ elements and the field for the root node), and therefore the SC of these vectors is $O(n)$. After this analysis of data structures used in Algorithm **??** we can state that the SC of the algorithm proposed is $O(max(n, m))$, i.e., it is linear.

Now considering a time complexity (TC) analysis we have that all functions implemented in the Algorithm **??** are atomic, i.e., they can be executed in $O(1)$. Then, to obtain the TC of Algorithm **??** we have to analyze the two main loops. The first loop (`for` in Line 2) is executed $n-1$ times while analyzing $n-1$ points. Although this loop uses a stack, the only insertion point into the stack CP is on the line 6. This fact confirms the SC $O(n)$ of the stack CP. Continuing on the loop presented in Line 2, we have an inner loop (`while`) guided by the stack CP in Line 10. This loop has an amortized TC at maximum $n-1$, since each time it is analyzed i.e., the nodes $[p_h, p_n]$, $[q_h, q_n]$ are compared) one edge will be inserted into the tree. The insertion will be done by either the loop itself (Line 13) or the conditions of the others two conditionals (when $p_h > q_h$ in Line 18 and when $p_h = q_h$ in Line 21). Then, the first loop has a linear TC,*i.e.*, i.e., $O(n)$. The second main loop (`while`) in Line 23 is executed whereas there are elements on the stack. Every time it is executed one element is removed from the stack. Thus this loop can be executed at maximum $n-1$ turns, i.e., the maximum stack size. Then, the second loop also has TC of $O(n)$. Therefore, the algorithm has a linear TC, i.e., $O(n)$.

3.4 Attributes

The component tree based approaches use measures extracted from the nodes of the tree structure. These measures are called attributes. Let $[h, n] \in \mathcal{C}(F)$. We define the height, the surface, and the volume attributes of the component $c_{h,n}$ as being:

$$\begin{aligned} ht(c_{h,n}) &= \max_{x \in c_{h,n}} \{F(x) - h_p\}, \\ s(c_{h,n}) &= \text{cardinality}(c_{h,n}), \\ v(c_{h,n}) &= \sum_{x \in c_{h,n}} (F(x) - h_p), \end{aligned}$$

respectively, where h_p is the parent weight of $c_{h,n}$. It is possible to compute, simultaneously, the component tree and these attributes of components without changing the time complexity of the algorithm.

4. Multithresholding

We now turn towards an application for the 1D component tree. Segmentation by multiple-threshold selection, or simply multithresholding, relies to the assumption that homogeneous regions present in the image can be detected in the histogram of the image. This segmentation method consists of selecting threshold levels by analyzing the histogram of the image. These thresholds determine histogram classes, and therefore any image pixel is classified according to the histogram class it belongs to.

In this work, we propose a method for multithresholding gray-level images in K levels. This method is based on the hypothesis that objects which appear on an image can be represented by salient classes present in the histogram of the image. These classes can be represented as the K most significative components extracted from component tree of the histogram of the image (an histogram is modelled as a WOS) - see Appendix for details.

The proposed method can be described in five main steps: 1) Histogram computation from gray-level image; 2) Computation of the Component tree of the histogram of the image; 3) Identification of the salient markers present in the histogram by means of the extraction of K most significative components of the histogram's component tree; 4) Histogram segmentation by watercourse transform (i.e., the dual of the watershed transform [1,3]) using the salient markers extracted in step 3; 5) Image segmentation by applying the segmented histogram to the original image.

Figure 3 shows the application of the proposed method to six classical images, namely: lena, goldhill, fruits, barbara, cameraman, and house. The first four images are of size 512×512 pixels and the last two of size 256×256 pixels. They are shown in the first column of Figure 3. We multithreshold all the histograms of image in five classes, and therefore the image in five levels. We chose the same number of classes for all images because all of them have at least five concise regions. On the other columns of Figure 3, we have, starting from the second column, the histograms of the input images, the five most important leaf components of the histograms of images and their not overlapped ascendant components coverage (as lighter as important - remark that the histograms are not smoothed), the segmented histograms in five classes (the classes are separated by vertical lines), and the output images with five levels (where the level for each histogram region was chosen as the nearest integer of the mean level in the respective histogram region).

We perform a quantitative comparison of our method, Kapur et al. [6], Khotanzad and Bouarfa [7], and Otsu [11] using a well-known objective measure, i.e., the Peak Signal to Noise Ratio (*PSNR*) [12]. The results are shown in Table 1. We observe that our method obtains *PSNR* values close to the *PSNR* value achieved by the other methods on four images: lena, barbara, cameraman and house. Indeed, we can see that our method segments the images in concise/homogeneous regions in 4 out of 6 images. However, in the goldhill and fruits images (second and third rows) the salient

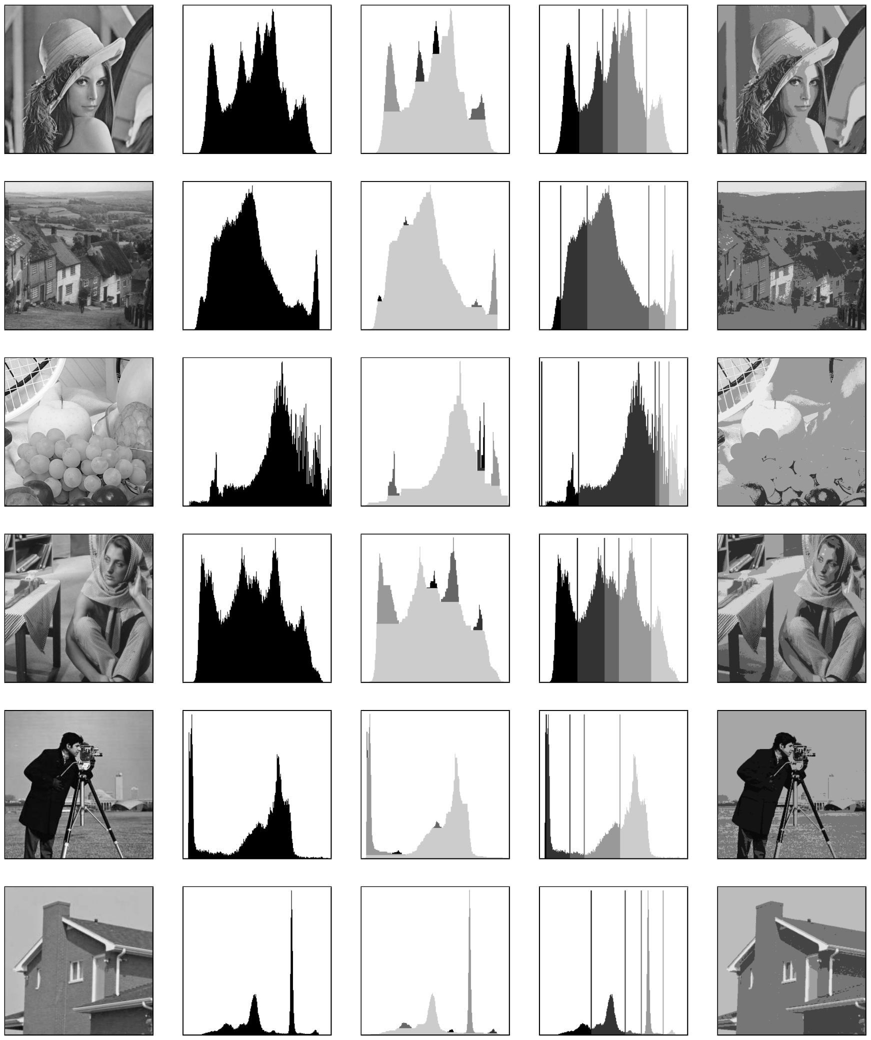


Figure 3. Real examples of classical images illustrating our multithresholding method. Columns from left to right: input original image, input image histogram, five (5) most important leaf components (maxima), segmented histogram in five (5) regions, and output segmented image in five (5) levels.

classes of the histograms of images are overlapped, and so our method is not suitable. In the cases where the histogram hypothesis holds, we argue that our method segments the images in regions more homogeneous than the other methods.

Table 1. PSNR for test images.

Images	Kapur	Khotanzad	Otsu	Our method
lena	25.3574	27.0722	28.2001	27.5316
goldhill	21.8978	22.4819	27.0583	21.6181
fruits	20.7996	22.5991	26.3987	19.6554
barbara	25.4540	26.1957	27.1348	26.4002
cameraman	19.3428	25.5831	27.8837	25.2907
house	20.1270	28.2576	29.3351	28.1030

5. Conclusion

In this paper we introduced, described, and illustrated a time and space linear complexity algorithm to compute the component tree for weighted ordered sets, i.e., 1D signals.

We proposed a new method for gray-level image multithresholding, based on the hypothesis that objects which appear on an image can be represented by salient classes present in a histogram of the image. These salient classes were modelled as the most significative components, where the importance corresponds to the volume attribute. Experiments showed that our method is competitive compared to classical ones when the hypothesis hold.

For future works, we plan to establish some methodology to select automatically the number of the most significative components present in the component tree, yielding an automatic multithresholding algorithm with respect to the number of classes in the output image. We also plan to extend our method to segment color images [4].

Acknowledgments

We would like to acknowledge support for this research from UFMG, CAPES/MEC, CNPq/MCT and FAPEMIG.

References

[1] S. Beucher and F. Meyer, *The morphological approach to segmentation: The watershed transform*, Mathematical Morphology in Image Processing, 1992, pp. 433–481.

[2] E. J. Breen and R. Jones, *Attribute openings, thinnings and granulometries*, Computer Vision and Image Understading **64** (1996), no. 3, 377–389.

[3] L. H. Croft and J. A. Robinson, *Subband Image Coding Using Watershed and Watercourse Lines of the Wavelet Transform*, IEEE Transactions on Image Processing **3** (1994), no. 6, 759–772.

[4] T. Geraud, P.-Y. Strub, and J. Darbon, *Color Image Segmentation based on Automatic Morphological Clustering*, IEEE ICIP (2001), pp. 70–73.

[5] R. Jones, *Component trees for image filtering and segmentation*, IEEE Workshop on Nonlinear Signal and Image Processing (1997).

[6] J. N. Kapur, P. K. Sahoo, and A. K. C. Wong, *A new method for Gray-Level Picture Thresholding Using the Entropy of the Histogram*, Computer Vision, Graphics, and Image Processing **29** (1985), 273–285.

[7] A. Khotanzad and A. Bouarfa, *Image Segmentation by a Parallel, Non-Parametric Histogram Based Clustering Algorithm*, Pattern Recognition **23** (1990), no. 9, 961–973.

[8] J. Mattes and J. Demongeot, *Efficient algorithms to implement the confinement tree*, DGCI (2000), LNCS, vol. 1953, pp. 392–405.

[9] J. Mattes, M. Richard, and J. Demongeot, *Tree representation for image matching and object recognition*, DGCI (1999), LNCS, vol. 1568, pp. 298–312.

[10] L. Najman and M. Couprie, *Building the component tree in quasi-linear time*, IEEE Transaction on Image Processing **15** (2006), no. 11, 3531–3539.

[11] N. Otsu, *A threshold selection method from grey-level histograms*, IEEE Transactions on Systems, Man and Cybernetics **9** (1979), no. 1, 41–47.

[12] M. Rabbani and P. W. Jones, *Digital Image Compression Techniques*, 1st, Society of Photo-Optical Instrumentation Engineers (SPIE), Bellingham, WA, USA, 1991.

[13] P. Salembier, A. Oliveras, and L. Garrido, *Anti-extensive connected operators for image and sequence processing*, IEEE Transaction Image Processing **4** (1998), no. 7, 555–570.

Appendix: Extracting the most significative components

In introduction, we have mentioned as simple use of the component tree the image filtering (removing nodes of the tree whose attribute value is below a given threshold). Here, we show a more advanced use for the component tree; determination of the K most significative components of the component tree. We hypothesized the volume attribute can model the importance of a salient region present in the histogram of the image. By using the tree, this task reduces to the search for the K nodes that have the largest attribute values and are not bound with each other (even transitively) by the inclusion relation. An algorithm to achieve this task is proposed in [10, Algorithm 3]. Its complexity is $O(sort(n) + n)$, where n is the number of points in the WOS and $sort(n)$ is the complexity of the sorting algorithm (it can be linear). Once the K most significative components are selected, we go back to the initial component tree and take as markers for the salient classes the leaf components corresponding to those K components. These markers are used in histogram segmentation by the watercourse transform.

Note that similar results could be obtained by performing attribute based operations using several volume threshold values.

A parallel implementation of the dual-input Max-Tree algorithm for attribute filtering

Georgios K. Ouzounis and Michael H. F. Wilkinson

Institute of Mathematics and Computing Science University of Groningen, The Netherlands
georgios@cs.rug.nl, m.h.f.wilkinson@rug.nl

Abstract This paper presents a concurrent implementation of a previously developed Dual-Input Max-Tree algorithm that implements anti-extensive attribute filters based on second-generation connectivity. The paralellization strategy has been recently introduced for ordinary Max-Trees and involves the concurrent generation and filtering of several Max-Trees, one for each thread, that correspond to different segments of the input image. The algorithm uses a Union-Find type of labelling which allows for efficient merging of the trees. Tests on several 3D datasets using multi-core computers showed a speed-up of 4.14 to 4.21 on 4 threads running on the same number of cores. Maximum performance of 5.12 to 5.99 was achieved between 32 and 64 threads on 4 cores.

Keywords: second-generation connectivity, Dual-Input Max-Tree, attribute filter, parallel computing, shared memory.

1. Introduction

Attribute filters [2, 9] are a class of shape preserving operators. Their key property is that they operate on image regions rather than individual pixels. This allows image operations without distorting objects, i.e., they either remove or preserve objects intact, based on some pre-specified property. Attribute filters can be efficiently implemented using the Max-Tree algorithm [9], or similar tree structures [3, 12]

Image regions in mathematical morphology are characterized by some notion of connectivity, most commonly 4- and 8-connectivity. This yields an association between connectivity and connected operators which is extensively discussed in [1, 8, 10]. These papers also provide extensions to these basic connectivities known as *second-generation connectivity.* A general framework and algorithm is presented in [7]. The algorithm referred to as the *Dual-Input Max-Tree* supports the mask-based connectivity scheme, for which we give a concurrent implementation in this paper. It is based on the parallel Max-Tree algorithm in [14], which builds individual Max-Trees for image regions concurrently, and merges these trees efficiently.

2. Attribute filters

Attribute filters are based on *connectivity openings.* In essence, a connectivity opening $\Gamma_x(X)$ yields the connected component containing the point $x \in X$ and $\emptyset$ otherwise. A connectivity opening is characterized by the following properties; for any two sets X, Y it is *anti-extensive* i.e., $\Gamma_x(X) \subseteq X$, *increasing* i.e., if $X \subseteq Y \;\Rightarrow\; \Gamma_x(X) \subseteq \Gamma_x(Y)$, and *idempotent* i.e., $\Gamma_x(\Gamma_x(X)) = \Gamma_x(X)$. Furthermore, for all $X \subseteq E$, $x, y \in E, \Gamma_x(X)$ and $\Gamma_y(X)$ are equal or disjoint.

A general approach in deriving second-generation connectivity openings using arbitrary image operators is given in [7]. A mask-based connectivity opening is defined as:

$$\Gamma_x^M(X) = \begin{cases} \Gamma_x(M) \cap X & \text{if } x \in X \cap M, & \text{(1a)} \\ \{x\} & \text{if } x \in X \setminus M, & \text{(1b)} \\ \emptyset & \text{otherwise.} & \text{(1c)} \end{cases}$$

where M is an arbitrary, binary mask image.

We can define a number of other connected filters based on a connectivity opening that work by imposing constraints on the connected components it returns. In the case of attribute openings such constraints are commonly expressed in the form of binary criteria which decide to accept or to reject components based on some attribute measure.

Attribute criteria Λ are put in place by means of a trivial opening Γ_Λ. The latter yields C if $\Lambda(C)$ is true, and $\emptyset$ otherwise. Furthermore, $\Gamma_\Lambda(\emptyset) = \emptyset$. Attribute criteria are typically expressed as:

$$\Lambda(C) = \mathit{Attr}(C) \geq \lambda, \tag{2}$$

with $\mathit{Attr}(C)$ some real-value attribute of C, and λ an attribute threshold.

Definition 1. The binary attribute opening Γ^Λ of a set X with an increasing criterion Λ is given by:

$$\Gamma^\Lambda(X) = \bigcup_{x \in X} \Gamma_\Lambda(\Gamma_x(X)). \tag{3}$$

$\square$

Many examples are given in [2, 9]. Note that if Λ is non-increasing we have an *attribute thinning* Φ^Λ [2] instead. An example is the scale-invariant non-compactness criterion of the form of (2), in which

$$\mathit{Attr}(C) = I(C)/V^{5/3}(C), \text{ where } I(C) = \frac{V(C)}{4} + \sum_{\mathbf{x} \in C} (\mathbf{x} - \overline{\mathbf{x}})^2, \tag{4}$$

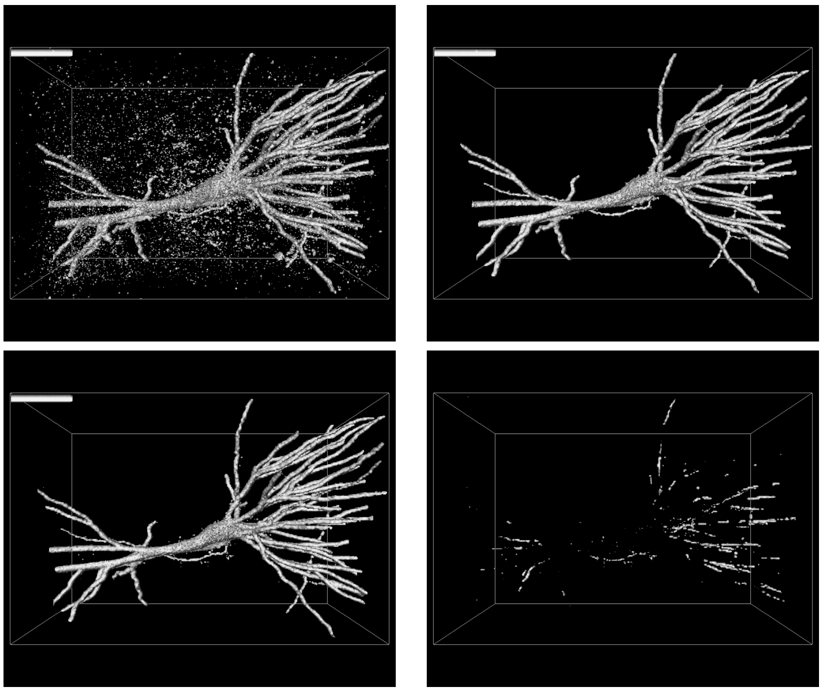


Figure 1. Isosurface projections of a confocal laser scanning micrograph of a pyramidal neuron and the output of the non-compactness filter (4) based on the 26-connectivity, both at isolevel 1. The first image in the bottom row illustrates the filter's performance using closing-based connectivity and the second shows the difference volumes between two attribute filter results. Various details within the neuron are lost using the 26-connectivity which are preserved by using a second-generation connectivity instead. See [7] for details.

with I the trace of the moment of inertia tensor in 3D and $V(C)$ the volume of a component C [15]. Attribute filters can be operated on sets characterized by second-generation connectivity by replacing Γ_x with Γ_x^M instead. The proof of this and a more detailed analysis can be found in [7]. Furthermore, an investigation in optimizing the parameters affecting the performance of these filters is discussed in [6] An example of attribute thinnings using closing-based second-generation connectivity is shown in Figure 1.

3. The Max-Tree algorithm

The Max-Tree was introduced by Salembier [9] as a versatile structure for computing anti-extensive attribute filters on images and video sequences. It is a rooted, unidirected tree in which the node hierarchy corresponds to the

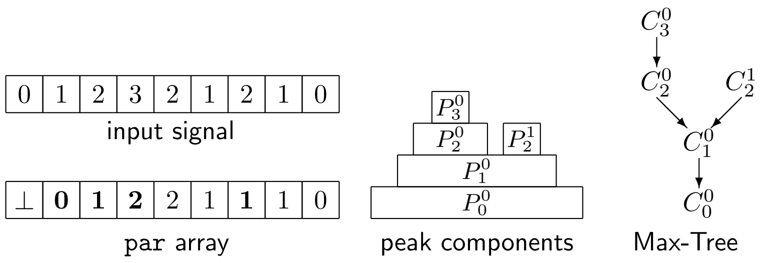


Figure 2. Example of input signal, peak components, Max-Tree and its encoding in a `par` array, in which $\perp$ denotes the overall root node, and boldface numbers denote the level roots, i.e., they point to positions in the input with grey level other than their own.

nesting of peak components given a gray-scale image. A peak component P_h at level h is a connected component of the thresholded image $T_h(f)$. Each tree node C_h^k (k is the node index) contains only those pixels of a given peak component which have gray-level h. In addition each node except for the root, points towards its parent $C_{h'}^{k'}$ with $h' < h$. The root node is defined at the minimum level h_{min} and contains the set of pixels belonging to the background.

The algorithm is a three-stage process in which the construction of the tree and the computation of node attributes is independent of filtering and image restitution. During the construction stage every pixel visited contributes to the auxiliary data buffer associated to the node it belongs to. Once a node is finalized, its parent inherits these data and recomputes its attribute. Inheritance in the case of increasing attributes such as area/volume is a simple addition while for non-increasing attributes such as the non-compactness measure of (4) the accumulation relies on more delicate attribute handling functions described in [7].

4. Including union-find in the Max-Tree

The hierarchical queue-based algorithm given by Salembier [9] cannot be trivially parallellized. In our approach we choose to partition the image into N_p connected disjoint regions the union of which is the entire image domain. Each region is assigned to one of the N_p processors for which a separate tree is constructed. The non-trivial part of this approach is the merging of the resulting trees. It is a process that requires (i) the merging of the peak components P_h^i, (ii) the updating of the parent relationships, and (iii) the merging of the attributes of the peak components. Parallellizing the filtering stage is trivial.

Previously, Najman et al. provided an algorithm to compute the Max-Tree using union-find [5]. Wilkinson et al. [14] use a different approach, using

Salembier et al.'s original algorithm [9] and changing the way the labels indicating node-membership of each pixel were chosen. Instead of using arbitrary numbers, Wilkinson et al. use the index of the first pixel of a node as the label. This means that each pixel of a node points to this "canonical element", which is referred to as a *level root.* The level root of a node itself is given the level root of its parent node as its index. These labels (or actually parent pointers in union-find terms) are stored in an array denoted `par`. Thus, if $f(\texttt{par}[x]) \neq f(x)$, x is a level root. In the algorithm in [14], after building a tree using a single thread, each $\texttt{par}[x]$ points directly to a level root: its own if x is not a level root, or to the level root of the parent node. An example is shown in Figure 2. Once the results of multiple threads are merged, this is no longer true. Therefore, we implement a function `levroot` to find the level root of any pixel. If $\texttt{levroot}(x) = \texttt{levroot}(y)$ x and y belong to the same node. The implementation of `levroot` also includes path compression as in [11].

5. The dual-input mode

As in the sequential case, the structure of the Max-Tree is dictated by the peak components of the mask volume m rather than the original volume f. An example is given in Figure 3. The dual-input version of the algorithm in [14] requires a number dummy nodes which assist in the merging of the different trees once all the threads return. To do this we double the size of the `par` array, and place the volumes f and m side by side in a single block of memory. In this way $f(p+volsize) = m(p)$ for all voxels p in the volume domain. For all p for which $f(p) \neq m(p)$ $\texttt{par}(p + volsize)$ will contain a valid reference to a level root.

The flooding function proceeds as described in [14] only we modify the way auxiliary data are handled and add a number of intensity mismatch checks to conform with the dual-input algorithm. After reaching a given level lev(=current level in mask m) and before retrieving any of the pixels available in the queue for that level, we first initialize the auxiliary data variable *attr*. It is set to the attribute count of the node corresponding to the $lero[lev]$. If an attribute count from a node at higher level is inherited through parameter *thisattr*, we update *attr*. A while loop then retrieves sequentially the members of the queue and for each one performs the mismatch check. If $f(p) \neq m(p)$ for a pixel p this signals the case in which p belongs to the current active node at $f(p)$ through the connected component at level $m(p)$, i.e., it defines a peak component at level $f(p)$ to which p in the mask volume is connected. In terms of our parallelizing strategy this means that it already defines a dummy node at $m(p)$ offset by *volsize*. We must then set $\texttt{par}(p + volsize)$ to $lero[lev]$. We must also create a new node at level $f(p)$ if none exists, and add p to the node at level $f(p)$. If $f(p) > m(p)$ p is a singleton (according to (1)). This requires finalizing the node which is done by setting its parent to $lero[lev]$, setting its auxiliary data to the

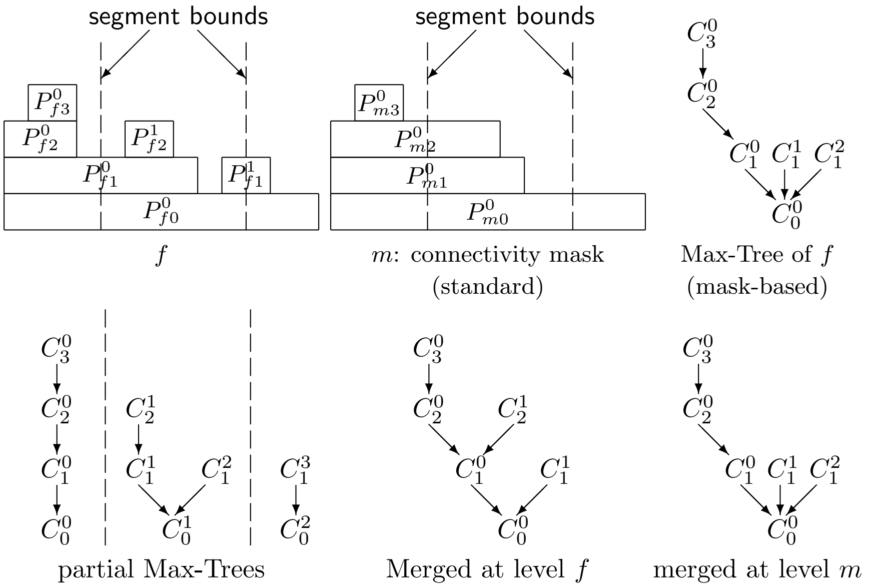

segment bounds
segment bounds
P_{f3}^0
P_{f2}^0
P_{f2}^1
P_{f1}^0
P_{f1}^1
P_{f0}^0
f
P_{m3}^0
P_{m2}^0
P_{m1}^0
P_{m0}^0
m: connectivity mask (standard)
C_3^0 C_2^0 C_1^0 C_1^1 C_1^2 C_0^0
Max-Tree of f (mask-based)
C_3^0 C_2^0 C_1^0 C_0^0 C_2^1 C_1^1 C_1^2 C_0^1 C_1^3 C_0^2
partial Max-Trees
C_3^0 C_2^0 C_2^1 C_1^0 C_1^1 C_0^0
Merged at level f
C_3^0 C_2^0 C_1^0 C_1^1 C_1^2 C_0^0
merged at level m

Figure 3. Dual-Input Max-Tree of 1D signal f using mask m: The attributes of C_2^0 and C_2^1 are merged to C_2^0 since all pixels at level $h = 2$ are clustered to a single peak component. Furthermore C_1^1 breaks up to a number of singleton nodes equal to the number of pixels in P_{f1}^1. Bottom row: partial Max-Trees of segments of signal indicated by the dashed lines; merger of partial Max-Trees at level of f at the boundaries yields standard Max-Tree in this case; merging at level of m at the boundary yields correct result.

unit measure and clearing $lero[f(p)]$. Details are given in Algorithm 1.

Otherwise, if $f(p) = m(p)$, it is necessary to check if the $lero[lev] \geq volsize$, i.e., if it is a dummy node. If this is the case, we update `par`$[lero[lev]]$ to p, and then set $lero[lev]$ to p, effectively setting the level root to a non-dummy node. The auxiliary data stored in $attr$ are then updated.

For every unprocessed neighbour q of p we determine where to create a new node. If $f(q) = m(q)$ the new node is q, otherwise $q + volsize$. If $lero[m(q)]$ exists, we set `par`$[newnode]$ to $lero[m(q)]$, otherwise $lero[m(q)]$ is set to `par`$[newnode]$. If $m(q) \geq lev$ we then enter into the recursion as in [9,14].

6. Concurrent merging of Max-Trees

As in regular connectivities, we must now connect the N_p Max-Trees. In [14], this is done by inspecting the pixels along the boundary between the parts, and performing the `connect` function on adjacent pixels on either

Algorithm 1 The flooding function of the concurrent Dual-Input Max-Tree algorithm.

```
procedure LocalTreeFlood($threadno, lero, lev, thisattr$) =
    Initialize auxilliary attribute data $attr$ and merge with $thisattr$
    while ($QueueNotEmpty(set, lev)$) do
        retrieve $p$ from queue at level $lev$
        if $f(p) \neq lev$ then
            par[$p + volsize$] := $lero[lev]$;
            if node at level $f(p)$ exists then
                add $p$ to it; par[$p$] := $lero[f(p)]$;
            else
                create node at level $f(p)$; $lero[f(p)] := p$;
            end;
            if $f(p) > lev$ then    (* singleton with parent at $lev$ *)
                finalize node; add $p$ to $attr$; par[$p$] := $lero[lev]$;
            end;
        else    (* No mismatch *)
            if $lero[lev] \geq volsize$ then    (* First pixel at level $lev$ *)
                par[$lero[lev]$] := $p$; $lero[lev] := p$;
            end;
            add $p$ to $attr$;
        end;    (* No mismatch *)
    end;    (* while *)
    for all neighbours $q$ of $p$ do
        if not $processed[q]$ then
            $processed[q] := true$; $mq := m(q)$;
            initialize $childattr$ to empty;
            if $m(q) \neq f(q)$ then $newnode := q + volsize$;
            else $newnode := q$; end;
            if $lero[m(q)]$ does not exist then $lero[m(q)] := newnode$;
            else par[$newnode$] := $lero[m(q)]$; end;
            while $mq > lev$ do
                $mq :=$ LocalTreeFlood($threadno, lero, mq, childattr$);
            end;
            add any data in $childattr$ to $attr$;
        end;
    end;    (* for *)
    detect parent of $lero[lev]$
    add auxilliary data in $attr$ to auxilliary data of $lero[lev]$
    set $thisattr$ to attribute data of $lero[lev]$
    return level of parent of $lero[lev]$
end LocalTreeFlood.
```

Algorithm 2 Concurrent construction and filtering of the Max-Trees, thread p.

```
process ccaf($p$)
    build dual input Max-Tree $Tree(p)$ for segment belonging to $p$
    var $i := 1$ , $q := p$ ;
    while $p + i < K \ \wedge \ q$ mod $2 = 0$ do
        wait to glue with right-hand neighbor ;
        for all edges $(x, y)$ between $Tree(p)$ and $Tree(p + i)$ do
            if $f(x) \neq m(x)$ then $x := x + volsize$;
            if $f(y) \neq m(y)$ then $y := y + volsize$;
            connect$(x, y)$ ;
        end ;
        $i := 2 * i$ ; $q := q/2$ ;
    end ;
    if $p = 0$ then
        release the waiting threads
    else
        signal left-hand neighbor ;
        wait for thread 0
    end ;
    $filter(p, lambda)$ ;
end ccaf.
```

side of the boundary. This function is shown in Algorithm 3. A proof of the correctness and a detailed discussion are given in [14]. The key reason why this works efficiently, is that merging two nodes containing x and y, with $f(x) = f(y)$ reduces to the assignment:

$$\texttt{par}[\texttt{levroot}(y)] := \texttt{levroot}(x). \tag{5}$$

This is easily verified as follows: `par`[`levroot`(y)] now points to a pixel with the same grey level because $f(x) = f(y)$, and `levroot`(x) = `levroot`(y) after assignment (5), so that x and y belong to the same node.

~~**Algorithm 3** Merging two Max-Trees.~~

Function `connect` is called by the process *concurrent construction and filter* or `ccaf`(see Algorithm 2), which corresponds to one of the threads of the concurrent merging algorithm. Each thread p first builds a Max-Tree for its own sub-domain V_p.

Process `ccaf` is called after initializing `par`, the auxiliary data functions and preparing the thread data. It starts off by first initializing the level root array *lero* and hierarchical queue for all gray-levels and finding the minimum voxel values in f and m. Having got the starting voxel of minimum grey

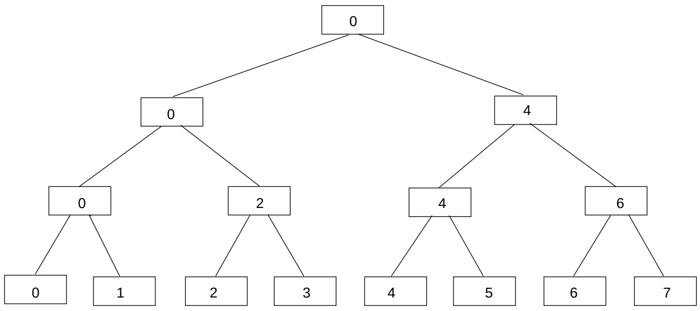


Figure 4. Binary tree used for merging domains.

value in m it calls `LocalTreeFlood`. If the minimum values in f and m differ, some post-processing as explained in [7] is required.

After this, the sub-domains are merged by means of a binary tree in which thread p accepts all sub-domains V_{p+i} with $p+i < N_p$ and $0 \leq i < 2^a$, where 2^a is the largest power of 2 that divides p. An example of a binary tree for $N_p = 8$ is shown in Figure 4. Note that odd-numbered threads accept no sub-domains. A thread that needs to accept the domain of its right-hand neighbor, has to wait until the neighbor has completed its Max-Tree computation. Because the final combination is computed by thread 0, all other threads must wait for thread 0 before they can resume their computation for the filtering phase. This synchronization is realized by means of two arrays of $N_p - 1$ binary semaphores. The filtering phase is also fully concurrent, and is identical to that described in [14].

For second-generation connectivity, the difference lies not in the implementation of `connect`, but in *which* pixels need to be merged. Suppose x and y are adjacent voxels which lie on different sides of the boundary inspected by `ccaf`. If $f(x) = m(x)$ the node in the Max-Tree at level $f(x)$ is the correct one, as before, otherwise we should start merging at level $m(x)$, as shown in Figure 3. At the left-hand segment boundary in this figure, merging at level $f(x)$ ignores the fact that P_{f2}^0 and P_{f2}^1 are clustered together in node C_2^0 using connectivity based on mask m. By contrast, at the right-hand segment boundary, merging from level $f(x)$ would merge nodes C_1^2 and C_1^3, which are considered singletons in the mask-based connectivity. In the scheme outlined above, this means that we start the merger from x if $f(x) = m(x)$, and from $x + volsize$, otherwise. The same holds for y. Thus the only changes to the `ccaf` function when compared to [14] lies in the statements immediately preceding the call to `connect`.

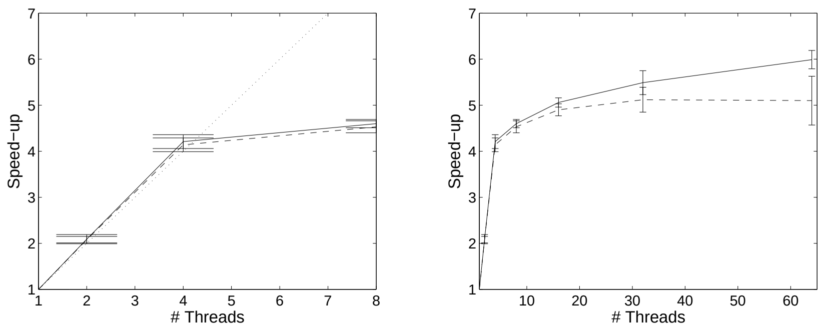


Figure 5. Speed-up for volume openings (solid) and non-compactness thinnings (dashed) as a function of number of threads. The left graph shows the initial, slightly better than linear (dotted-line) speed-up as we move from 1 to 4 threads. The right-hand graph also shows the behaviour up to 64 threads.

7. Performance testing and complexity

The above algorithm was implemented in C for the general class of anti-extensive attribute filters. Wall-clock run times for numbers of threads equal to 1, 2, 4, 8, 16, 32, and 64 for for two different attributes were determined. The attributes chosen were volume (yielding an attribute opening) and the non-compactness measure (4) [15] yielding an attribute thinning.

Timings were performed on an AMD dual-core, Opteron-based machine. This machine has two dual-core Opteron 280 processors at 2.4 GHz, giving a total of 4 processor cores, and 8 GB of memory (4 GB per processor socket). Each timing was repeated 10 times, and the minimum was used as the most representative of the algorithm's performance. Five volume data sets publicly available from `http://www.volvis.org` were used. All volumes were 8 bit/voxel sets, comprising 4 CT-scans and 1 MRI scan. Test were done using volume openings with $\lambda = 100$ and φ_1 with $\lambda = 2.0$ and the subtractive rule. The volume sizes ranged from 22.7 to 128 MB. The speed-up achieved is shown in Figure 5. As can be seen, the speed-up is slightly better than linear, as we move from 1 to 4 threads (4.21 $\pm$0.15 for volume openings and 4.14 $\pm$ 0.15 for non-compactness thinning at 4 threads). This may be due to the fact that more than 4 GB of memory is required when processing the larger volumes in the set, and therefore the processor doing the work requires access to the memory bank of the other socket, resulting in higher latency. As the number of threads exceeds the number of cores, we still obtain more speed-up, up to 5.99 $\pm$ 0.2 at 64 threads for volume openings, and 5.12 $\pm$ 0.27 at 32 threads for non-compactness thinning. In absolute terms, computing time went from between 20.8 and 128 s down to between 4.66 and 23.4 s.

The complexity of the algorithm is governed by two main parts: the building phase and the merging phase. Assuming a volume of $X \times Y \times Z = N$, in the building phase the time complexity is $O(GN/N_p)$, with G the number of grey levels, and N_p the number of processors. This complexity arises from the $O(GN)$ complexity of Salembier et al.'s Max-Tree algorithm [4]. If the number of grey levels is large, it may be better to replace this by Najman and Courpie's method [5]. The merging phase has complexity $O(GXY \log N \log N_p)$ if the volume is split up into slices orthogonal to the Z direction. The $\log N$ is due to the fact that we only use path compression, not union-by-rank. Memory requirements are $O(N + G)$.

8. Conclusions

The speed-up of the algorithm presented is similar to that of the parallel Max-Tree algorithm in [14]. However, it is about 50% slower in absolute terms. The speed-up if the number of threads exceeds the number of physical processors is due to reduced cache thrashing, as is confirmed by profiling. It also indicates that on machines with more processing cores, a (near) linear speed up beyond 4 CPUs is expected.

Apart from use in 3D data, the algorithm could be of use in the efficient implementation of attribute-space connected filters [13], in which the 2D input image is embedded into a higher-dimensional attribute space, followed by application of a connected filter in that space.

Given the ready availability of multi-core processors, this algorithm is not restricted to supercomputers anymore, but will be of use to many, and in the near future most desktop machines.

References

[1] U. M. Braga-Neto and J. Goutsias, *A theoretical tour of connectivity in image processing and analysis*, J. Math. Imag. Vis. **19** (2003), 5–31.

[2] E. J. Breen and R. Jones, *Attribute openings, thinnings and granulometries*, Comp. Vis. Image Understand. **64** (1996), no. 3, 377–389.

[3] R. Jones, *Connected filtering and segmentation using component trees*, Comp. Vis. Image Understand. **75** (1999), 215–228.

[4] A. Meijster and M. H. F. Wilkinson, *A comparison of algorithms for connected set openings and closings*, IEEE Trans. Pattern Anal. Mach. Intell. **24** (2002), no. 4, 484–494.

[5] L. Najman and M. Couprie, *Building the component tree in quasi-linear time*, IEEE Trans. Image Proc. **15** (2006), 3531–3539.

[6] G. K. Ouzounis and M. H. F. Wilkinson, *Filament enhancement by non-linear volumetric filtering using clustering-based connectivity*, Proc. iwicpas 2006, Unknown Month 2006Aug 26, pp. 317–327.

[7] ———, *Mask-based second-generation connectivity and attribute filters*, IEEE Trans. Pattern Anal. Mach. Intell. **29** (2007), 990–1004.

[8] C. Ronse, *Set-theoretical algebraic approaches to connectivity in continuous or digital spaces*, J. Math. Imag. Vis. **8** (1998), 41–58.

[9] P. Salembier, A. Oliveras, and L. Garrido, *Anti-extensive connected operators for image and sequence processing*, IEEE Trans. Image Proc. **7** (1998), 555–570.

[10] J. Serra, *Connectivity on complete lattices*, J. Math. Imag. Vis. **9** (1998), 231–251.

[11] R. E. Tarjan, *Efficiency of a good but not linear set union algorithm*, J. ACM **22** (1975), 215–225.

[12] L. Vincent, *Granulometries and opening trees*, Fundamenta Informaticae **41** (2000), 57–90.

[13] M. H. F. Wilkinson, *Attribute-space connectivity and connected filters*, Image Vis. Comput. **25** (2007), 426–435.

[14] M. H. F. Wilkinson, H. Gao, W. H. Hesselink, J. E. Jonker, and A. Meijster, *Concurrent computation of attribute filters using shared memory parallel machines*, IEEE Trans. Pattern Anal. Mach. Intell. (2007). Submitted.

[15] M. H. F. Wilkinson and M. A. Westenberg, *Shape preserving filament enhancement filtering*, Proc. miccai'2001, 2001, pp. 770–777.

Multi-level decomposition of Euclidean spheres

Michael S. Vaz[1], Atilla P. Kiraly[2] and Russell M. Mersereau[3]

[1] *Barco Medical Imaging Systems, Beaverton, OR, USA*
`michael.vaz@barco.com`

[2] *Siemens Corporate Research, Princeton, NJ, USA*
`atilla.kiraly@siemens.com`

[3] *Georgia Institute of Technology, Atlanta, GA, USA*
`rmm@ece.gatech.edu`

Abstract Mathematical Morphology (MM) is used in medical image analysis for applications such as segmentation and skeletonization. The available efficient MM methods do not readily adapt to true Euclidean disc/sphere structuring elements (SE), which we are particularly interested in, without sacrificing accuracy for efficiency. An efficient method for MM using convex/symmetric SE, including Euclidean discs/spheres, is presented. Performance results for the proposed method are compared to the performance of a commercially available software package for a $512 \times 512 \times 418$ chest CT dataset. Increasing gains of the method for larger SE are demonstrated, making the method suitable for analysis of high resolution images. The method is efficient for iso/anisotropic SE.

Keywords: structuring element decomposition, sphere, disc, circle.

1. Introduction

Mathematical morphology (MM) is based on set theory and can be used to analyze image shape features [5, 7, 15]. MM is useful for a wide variety of applications including object recognition, image segmentation, and industrial inspection [16]. In medical imaging, MM is used for such applications as brain segmentation from MR images [4] and airway/vessel segmentation from CT images [6, 10].

Dilation and erosion are the elementary operations of MM. Other operations such as morphological opening and closing may be formed by combining dilation and erosion in sequence [15, 16]. A structuring element (SE) is the morphological kernel that is translated over the image domain and compared with the overlapping image region. For flat (binary) SE, the comparison operation is a local maximum for dilation and a local minimum for erosion, where the local neighborhood is defined by the SE's shape. These min/max comparisons for gray-level images reduce to logical AND/OR operations for erosion/dilation of binary images using flat SE.

Morphological analysis methods used in medical imaging generally require SE of different sizes and shapes. Since some of these methods require multiple larger sized SE, which require greater processing time, such methods can take several hours for a single image [10]. Long processing times impede clinical utility of applications as well as application/algorithm development itself.

The processing time for brute-force implementation of MM is proportional to the discrete mass (number of pixels/voxels) of the SE. As such, for 2D and 3D SE, which are used for medical image processing, the processing time can increase polynomially with respect to SE diameter and can become prohibitive. A multitude of methods have been proposed to accelerate MM and the majority of these involve some form of SE decomposition. The HGW algorithm is regarded as the most efficient for a 1D straight line SE [18, 19]; logarithmic decomposition (LD) is also very efficient for 1D and as such may be applied to separable 2D and 3D SE [3, 5]. However, only a subset of useful SE are separable. Some methods are applicable only to binary images or require preprocessing/encoding of the image [9, 12]; others cater to particular constraints such as the limited region of support of specialized hardware [8, 13].

We are particularly interested in 2D Euclidean disk and 3D Euclidean sphere SE that incur no further approximation beyond quantization itself, with no restriction placed on the isotropy of the quantization. Efficient methods that have already been proposed for convex/symmetric SE [11, 21] are usually discussed and demonstrated in 2D and don't readily lend themselves to a true 3D extension. Although the sphere falls into convex/symmetric category, as far as we know, these methods may not be used to obtain the decomposition of a sphere. Moreover, the definition of convexity used in these methods is not consistent and as such, Euclidean disks/spheres are outside the scope of some such methods. Methods for decomposition of a disc SE usually sacrifice accuracy for efficiency and still do not lend themselves to the 3D sphere [1, 5, 19, 21]. Some methods refer to gray-level disk SE as spheres [1, 5]; these SE are not the focus of this paper and are not to be confused with binary 3D spheres. Although the method using local histograms described in [5] may be applied to spheres, it does not readily lend itself to an efficient vectorized implementation, which is important for certain types of computation platforms.

SE decomposition as a union of partitions is described in [2] where genetic algorithms are used to compute the decomposition. A deterministic method for decomposing a 2D disc SE into a union of partitions is introduced in [20]. However, the decomposition is described analytically and may be difficult to understand and implement. We present a reworking of this method and describe the decomposition using morphological primitives. As such, it is now easier to understand/implement. The new morphological decomposition naturally extends itself to all 2D and 3D convex symmetric SE, such as Euclidean spheres. We present timing results and comparison for a large CT lung volume dataset.

In the remainder of this paper we refer to the following notations and definitions and introduce additional elements as necessary.

$\cup : Union$	$\oplus$: *Dilation*	$\bullet : Morphological\ closing$
$\cap : Intersection$	$\ominus$: *Erosion*	$\circ : Morphological\ opening$

Definition 1 (Convex SE)**.** If the discrete SE is equal to the set of all voxels that fall inside its Euclidean convex hull [17] it is considered to be convex. Continuous domain convex shapes that are discretized satisfy this definition of digital convexity. □

Definition 2 (Symmetric SE)**.** A symmetric SE has a clearly defined center. When the center of a symmetric SE is translated to the origin of a standard Cartesian coordinate system, the translated SE would be symmetric about the $x = 0, y = 0$, and $z = 0$ planes. □

Definition 3 (Sparse SE)**.** In general, a 3D SE that is not face-connected, or a 2D SE that is not edge-connected, is said to be sparse. □

2. Method

2.1 Overview of the proposed decomposition

The proposed method decomposes the SE into a union of partitions (Equation 1a), where each partition, P_i, consists of two factors; the first is the largest cube, C_i, that can morphologically open the partition without change, and the second is a sparse factor S_i (Equation 1b). This method will decompose any 2D or 3D convex/symmetric SE.

$$SE = P_1 \cup P_2 \cup P_3 \dots \tag{1a}$$
$$SE = (C_1 \oplus S_1) \cup (C_2 \oplus S_2) \cup (C_3 \oplus S_3) \dots \tag{1b}$$
$$SE = (C_1 \oplus S_1) \cup ((C_1 \oplus L_2) \oplus S_2) \cup ((C_1 \oplus L_2 \oplus L_3) \oplus S_3) \dots \tag{1c}$$
$$SE = C_1 \oplus (S_1 \cup L_2 \oplus (S_2 \cup (L_3 \oplus S_3) \dots)) \tag{1d}$$

As a consequence of convexity of the underlying SE, C_1 is a factor of C_2 and C_2 is a factor of C_3 and so on (Equation 1c). The L_i factors that relate the different C_i factors are described below and illustrated in Figure 3. This allows us to further decompose C_i and reuse morphological comparisons across the cubic factors (Equation 1d).

Figure 1 illustrates Equation 1a and shows the three partitions obtained by using the proposed method to decompose the example 2D SE shown on the right hand side. The cubic and sparse factors of each of these partitions are shown in Figure 2, which illustrates Equation 1b.

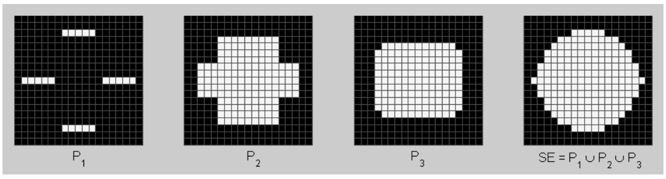


Figure 1. The proposed method decomposes the example SE (right) into a union of partitions P_1, P_2, and P_3. The origin of each partition is the center. The partitions overlap; this is possible due to the idempotency property of comparison operations (OR, AND, min, max) used to combine the partitions.

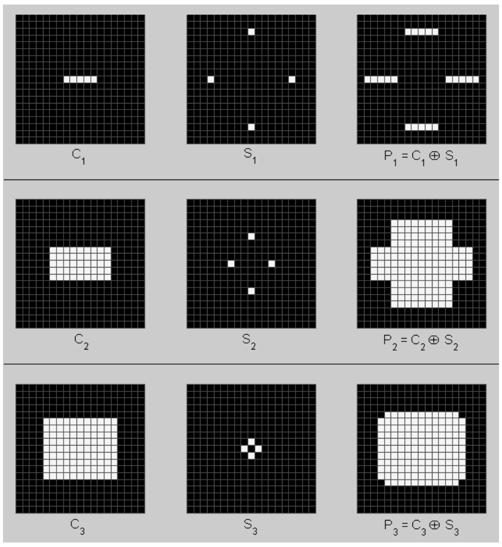


Figure 2. Each partition P_i has a cubic factor C_i and a sparse factor S_i, where $i = 1, 2, 3$ in this example. The origin of each factor is at its center.

2.2 Usage of the decomposition

Binary image dilation and erosion

$$I \oplus SE = (I \oplus P_1) \cup (I \oplus P_2) \cup (I \oplus P_3) \ldots \tag{2a}$$

$$I \oplus SE = (I \oplus C_1 \oplus S_1) \cup (I \oplus C_2 \oplus S_2) \cup (I \oplus C_3 \oplus S_3) \ldots \tag{2b}$$

$$I \oplus SE = I \oplus C_1 \oplus (S_1 \cup L_2 \oplus (S_2 \cup (L_3 \oplus S_3) \ldots)) \tag{2c}$$

Substituting $\oplus$ with $\ominus$ and $\cup$ with $\cap$ yields the equations for erosion.

Gray-level image dilation and erosion

$$I \oplus SE = \text{Supremum}((I \oplus P_1), (I \oplus P_2), (I \oplus P_3) \ldots) \tag{3}$$

$$I \ominus SE = \text{Infimum}((I \ominus P_1), (I \ominus P_2), (I \ominus P_3) \ldots) \tag{4}$$

Efficient MM for the cubic factors C_i

The cubic factors of each partition may be decomposed further. Logarithmic decomposition (LD) [5] is simple and can yield efficiency through computation reuse across partitions. The logarithmic factors for the considered example are illustrated in Figure 3. The figure shows how the dilation or erosion for all three cubic factors can be obtained with merely 9 comparison operations (ops) per output pixel or voxel. This can be accomplished by using a cascaded implementation as is suggested in Equation 2c.

While we favor the cascaded implementation with LD as described above, the HGW algorithm [18] may be used to efficiently compute the morphology result for the cubic factors. However this algorithm is more complex and does not gain from computation reuse that is possible due to the fact that C_i is a factor of C_{i+1}. If the user prefers parallelization, using the proposed decomposition in the form shown in Equation 2b with the HGW algorithm might be a good choice.

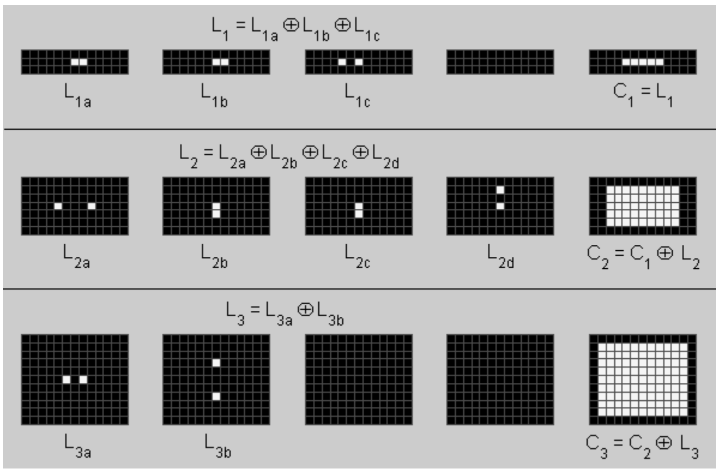


Figure 3. C_1 is a factor of C_2 and C_2 is a factor of C_3; as such, we may reuse computation across cubic factors. We use logarithmic decomposition (LD) to simply and efficiently perform the dilation/erosion of Ci. The origin of each C_i should be in its center. This can be satisfied either by assigning the origin of each logarithmic factor L_{ix} accordingly, or by updating the origin on each C_i. This is possible due to translational invariance property of dilation.

2.3 Decomposing a SE using the proposed method

Figure 4 illustrates the steps for decomposing a SE. The proposed decomposition follows the steps below.

1. Initialization: set CSE (current SE) to be equal to SE.
2. Find the largest cube with which CSE might be morphologically opened without change. This is the cubic factor C_i.

3. Find the corresponding sparse factor S_i. (See discussion below and Figure 5). Now that C_i and S_i have been found we have essentially decomposed a partition from CSE.
4. Update CSE with the subset that remains to be decomposed (RSE: remaining SE) as shown in Figure 4.
5. Go to Step 2 and repeat until RSE is NULL.

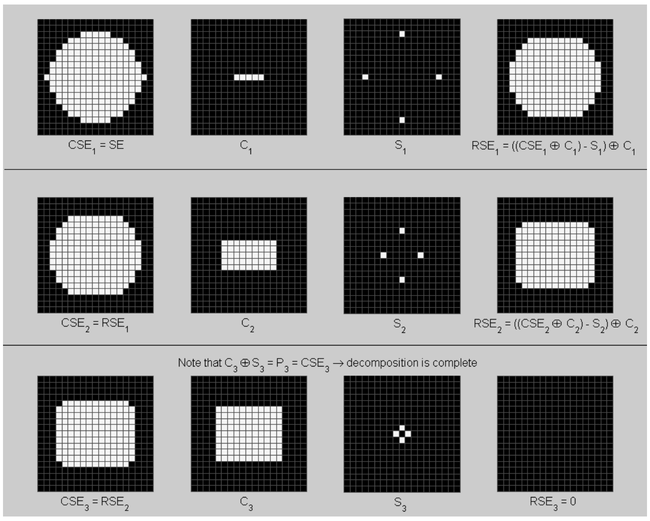

$CSE_1 = SE$
C_1
S_1
$RSE_1 = ((CSE_1 \oplus C_1) - S_1) \oplus C_1$
$CSE_2 = RSE_1$
C_2
S_2
$RSE_2 = ((CSE_2 \oplus C_2) - S_2) \oplus C_2$
Note that $C_3 \oplus S_3 = P_3 = CSE_3 \rightarrow$ decomposition is complete
$CSE_3 = RSE_2$
C_3
S_3
$RSE_3 = 0$

Figure 4. Decomposing the SE. At each iteration the current SE (CSE) is updated to reflect the subset of the SE that remains to be decomposed (RSE). C_i is the largest cube with which CSE_i can be morphologically opened without change. We can then determine S_i which completes the task of determining P_i and then determine the subset of CSE_i that remains to be decomposed RSE_i. The next iteration begins with updating CSE_{i+1} by setting it equal to RSE_i.

Determining S_i

Figure 5 illustrates the process to determine S_2 for the example in Figures 1–4.

1. Once C_i is available, obtain candidate sparse factors ST_1, ST_2, ST_3.
2. Test each ST_j for sparseness (see sub-section below).
3. Select the ST_j that is sparse and has the largest number of voxels (largest discrete mass) and assign it to S_i.

Testing for sparseness

We enforce that our sparse factor is not face-connected for 3D. For 2D, it should not be edge-connected. There are a few exceptions to this rule, such as if there is connectivity with the central voxel (local origin) during the first iteration of decomposition, i.e., during the process of determining S_1.

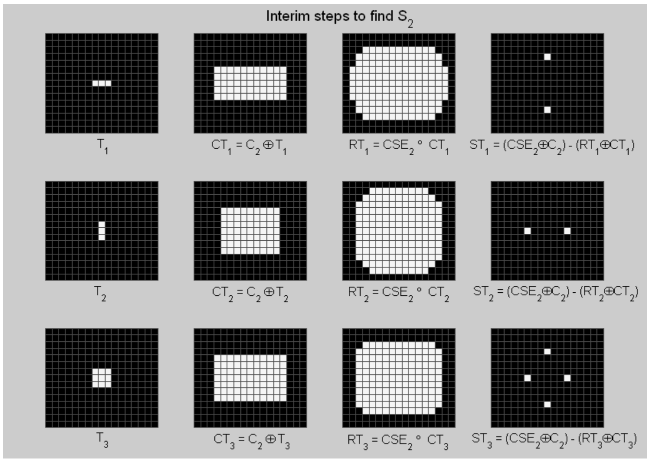

Interim steps to find S_2
T_1
$CT_1 = C_2 \oplus T_1$
$RT_1 = CSE_2 \circ CT_1$
$ST_1 = (CSE_2 \oplus C_2) - (RT_1 \oplus CT_1)$
T_2
$CT_2 = C_2 \oplus T_2$
$RT_2 = CSE_2 \circ CT_2$
$ST_2 = (CSE_2 \oplus C_2) - (RT_2 \oplus CT_2)$
T_3
$CT_3 = C_2 \oplus T_3$
$RT_3 = CSE_2 \circ CT_3$
$ST_3 = (CSE_2 \oplus C_2) - (RT_3 \oplus CT_3)$

Figure 5. Determining S_2 for the considered 2D example. Obtain test cubic factors CT_j, by dilating C_i with T_j, where j = 1, 2, and 3 for 2D SE. As illustrated above, we can then obtain a set of candidate RSE, RT_j, and candidate sparse factors ST_j. Determine the subset of ST_j that is indeed sparse and from this subset pick the "best" one and assign it to S_i. We apply a greedy criterion for "best", which is to pick the sparse ST_j that has the largest number of pixels/voxels. The implementation tests each ST_j for sparseness. Of course, once a particular ST_j is selected, we can simply assign its corresponding RT_j to RSE_i.

2.4 Method for 3D SE

- Once the decomposition is available for a 3D SE, the usage is identical to the 2D case discussed above.
- Decomposition is slightly more elaborate in the stage of selecting a sparse factor. Whereas for the 2D case we have three ST_j sparse factor candidates to consider, for the 3D case we have seven. This is a consequence of now having T_j, where $j = 1, 2, \ldots 7$. These T_j are the bar-3 and the square-3, each in the horizontal, vertical and axial orientations as well as the cube-3 (cube with discrete diameter 3).

3D sphere example

Figures 6, 7 and 8 illustrate the cascaded implementation of the proposed method for a radius-5.5 Euclidean sphere obtained under isotropic unit quantization. The method decomposes this SE into 3 partitions, as illustrated in the figures. Only the upper half of the SE is illustrated without loss of generality due to symmetry. The cubic factors C_1, C_2, C_3 are equal to cube-3, cube-5, cube-9 respectively. The origin is marked in dark gray (center of slice 6 of 11) in each of the figures and is not part of S_1 or S_2, which are depicted in medium gray; S_3, however, is just the single voxel at the origin. The medium and light gray voxels together depict a particular partition. The gray and white voxels together in Figure 7 illustrate the

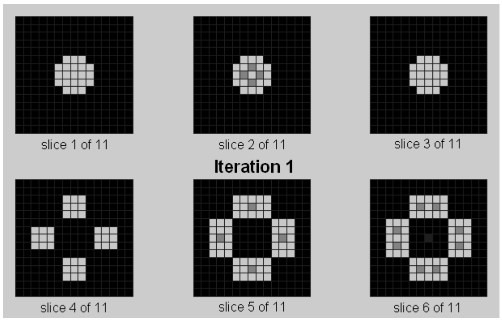


Figure 6. Upper half of P_1 for a radius-5.5 Euclidean sphere at unit quantization. The origin is the dark gray voxel in the center of slice 6, P_1 is the union of the light and medium gray voxels, S_1 is medium gray. C_1 is cube-3.

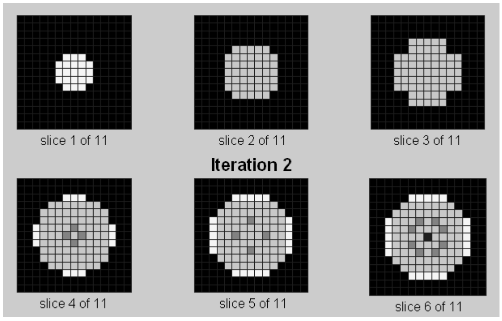


Figure 7. Foreground is $P_1 \cup P_2$ for the radius-5.5 sphere (only upper half of the sphere is shown). The origin is dark gray, P_2 is the union of light and medium gray voxels, S_2 is medium gray, the region of P_1 that does not overlap with P_2 is white. C_2 is cube-5.

union of P_1 and P_2, where the white pixels indicate the portion of the SE that is exclusively covered by P_1. The union of the gray and white voxels in Figure 8 depicts the union of P_1, P_2 and P_3 and is thus the SE itself. The white voxels in Figure 8 indicate the region that is covered exclusively by $P_1 \cup P_2$.

3. Results and analysis

The proposed method was implemented and compared to the commercially available SDC morphology toolbox [14]. Dilation on a gray-level chest CT image, sized $512 \times 512 \times 418$, was performed using various spherical SE. The standard "brute-force" direct implementation was also compared. Timing results were obtained for spherical SE with discrete radii 1-12. A dual-CPU Intel 2.0 GHz *Xeon*™ computer with 2 GB of RAM was used. The proposed method was implemented as a single-threaded executable. As is evident in Figure 9, the processing time for our own implementation of the

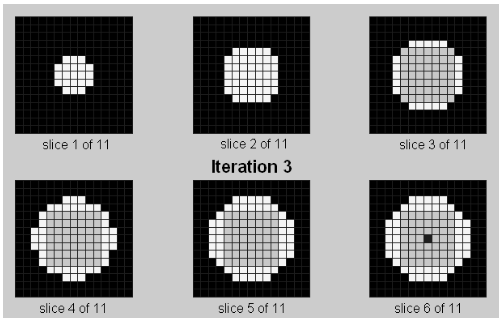


Figure 8. The origin coincides with S_3 and is dark gray, P_3 is the union of light gray and the origin. C_3 is cube-7, which coincides with P_3. The region of $P_1 \cup P_2$ that does not overlap with P_3 is white. The union of all foreground voxels is the union of all three partitions and is the radius-5.5 Euclidean isotropic sphere.

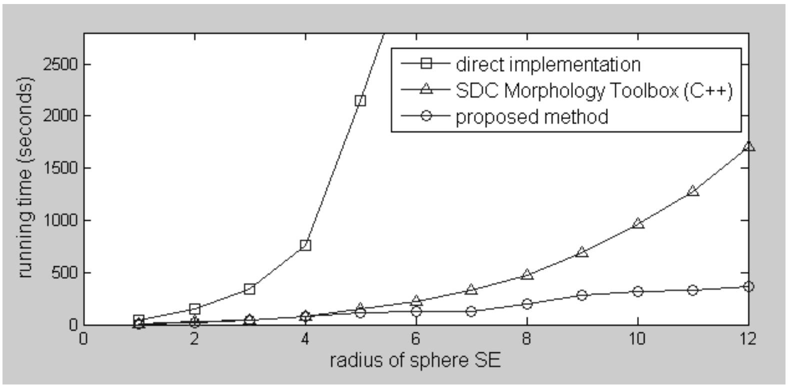


Figure 9. Processing times for gray-level dilation (binary SE) using sphere SE for a 512 × 512 × 418 chest CT image. Results obtained on and Intel Pentium IV Xeon Dual CPU 2.00 GHz platform with 2.00 GB of RAM.

direct method becomes prohibitive for larger SE. The algorithmic advantage of the proposed method facilitates gains over the SDC Morphology Toolbox that increase as SE size increases.

In [13], the four factors on the right of Figure 10 are offered as an optimal decomposition for the convex/symmetric SE shown to their left ("Iteration 2"). The proposed method will decompose the SE into 2 partitions whose cubic factors are square-3 and square-7 and obtain the MM result in 13 ops per output pixel (4 ops for C_1, 4 ops for L_2, and 5 ops for the sparse factors and the union), while the 4 factor decomposition requires 16 ops.

The number of comparison ops required per output voxel for dilation or erosion using discrete Euclidean sphere SE of radii 1-40 was calculated for the direct implementation, the proposed method, and two variations of the proposed method as shown in Figure 11. The op count for the direct implementation is one less than the number of voxels in the SE. One flavor

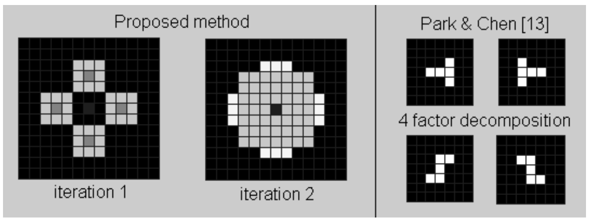


Figure 10. [Left] 2-partition decomposition of a 2D SE using the proposed method. Cubic factors are square-3 and square-7. Dark gray - origin (center): not part of S_1, however S_2 is the pixel at the origin. Light and medium gray together - the partition, medium gray (only in iteration 1) - S_1, white - region of P_1 that does not overlap with P_2. [Right] 4-factor decomposition for the same SE from Example 1 in [13]. The proposed method is more efficient.

of the proposed method was to consider the sphere SE as the collection of its 2D slices, implement the MM for each of the unique slices separately using the proposed method and combine the results. This implementation is less efficient and requires more memory. It illustrates that a true 3D method is superior to a 2D method directly applied to 3D SE, as would be required in order to make use of most of the applicable methods available in the literature.

The majority of comparison ops of the proposed method occur by way of the sparse factors of the partitions: for a radius-20 sphere SE, the proposed 2-canvas implementation (Equation 2c) would incur 600 comparisons per output voxel, 36 of which would be due to the cubic factors while the remaining 564 would be due to the sparse factors. By decomposing the sparse factors, this could be reduced to 292, thus reducing the total number to 328 ops per output voxel. While this could be considered an optimization, it comes at the expense of a third copy of the image volume for interim work, which may be undesirable for computation platforms with memory capacity constraints.

4. Discussion and conclusion

An efficient method for MM using Euclidean disk and sphere SE has been presented. These SE do not lend themselves well to factorization methods that can usually be used for convex/symmetric SE. The proposed method can be applied to all 2D/3D convex/symmetric flat (binary) SE. The method may be used for binary as well as gray-scale images and does not require any prior analysis or encoding of the image: its gain is not based on image content. The low memory overhead and efficient looping as well as vectorized implementations promote the utility of the method for development (scripting) as well as application platforms.

We believe the challenge of decomposing SE into partitions to be an instance of an NP-complete problem. The proposed method uses locally optimal (greedy) criteria to select a sparse factor during each iteration of

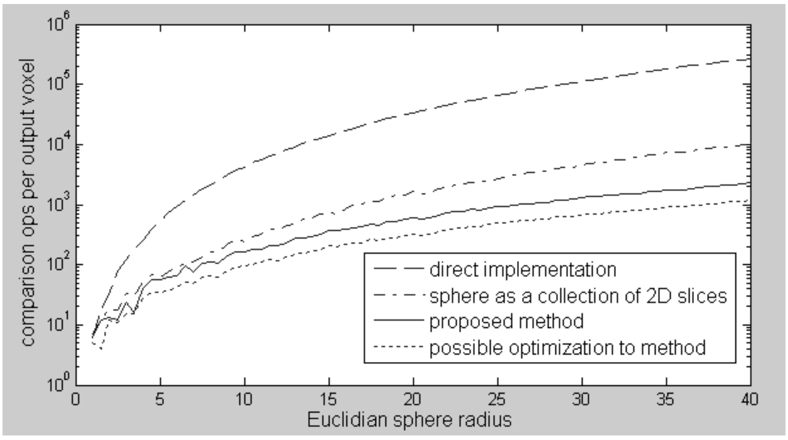


Figure 11. Number of comparison ops required per output voxel for MM using sphere SE. Note, op count presented on a log scale. Data obtained analytically. Proposed implementation follows Equation 2c and implements the sparse factors directly. A possible optimization would be to decompose the sparse factors, but this would require another interim copy of the image volume.

the decomposition. As such, it is not guaranteed to be optimal. We are investigating optimality relevant to the proposed method and expect to report on how the proposed greedy method compares to an optimal decomposition. This will of course require a clear definition of optimality.

The method is robust to scale: it works for iso/anisotropic SE. As such, it facilitates the development of scale-robust applications/algorithms. For medical image segmentation, SE may be defined in the context of anatomy instead of using discrete parameters. The chest CT image used to obtain the results shown in Figure 5 was quantized to a *mm* scale of $0.67 \times 0.67 \times 0.80$. With the proposed method, an algorithm could define a 3 *mm* sphere SE and this SE would be discretized according to the quantization scale of the CT image at run-time.

A comparison of the proposed method with others is underway. An empirical assessment of the gains of the method in the context of real medical applications will be performed. Since the method lends itself to vectorization, even hardware acceleration by means of a graphics processing unit (GPU) implementation would be possible.

References

[1] R. Adams, *Radial decomposition of disks and spheres*, Computer Vision, Graphics, and Image Processing **55** (1993), no. 5, 325–332.

[2] G. Anelli and A. Broggi, *Decomposition of arbitrary shaped binary morphological structuring elements using genetic algorithms*, IEEE Trans. Pattern Analysis and Machine Intelligence **20** (1998), no. 2, 217–224.

[3] D. Bloomberg, *Implementation Efficiency of Binary Morphology.* Available from: <`http://www.leptonica.com`> Access in: 2006-01-05.

[4] B. Dogdas, D. W. Shattuck, and R. M. Leahy, *Segmentation of the skull in 3D human MR images using mathematical morphology*, SPIE Medical Imaging (San Diego, USA, February 23–28, 2002) (M. Sonka and J. M. Fitzpatrick, eds.), Vol. 4684, SPIE, Bellingham, 2002, Medical Imaging 2002: Image Processing, pp. 1553–1562.

[5] V. Droogenbroeck and H. Talbot, *Fast computation of morphological operation with arbitrary structuring elements*, Pattern Recognition Letters **17** (1996), no. 14, 1451–1460.

[6] S. Eiho and Y. Qian, *Detection of coronary artery tree using morphological operator*, SPIE Medical Imaging (Lund, Sweden, September 7–10, 1997), Computers in Cardiology 1997 (A. Murray and S. Swiryn, eds.), Vol. 24, IEEE, Piscataway, 1997, 1997, pp. 525–528.

[7] R. Gonzalez and R. Woods, *Digital Image Processing*, 1st ed., Addison-Wesley Publishing Company, Reading, 1992.

[8] R. F. Hashimoto and J. Barrera, *A greedy algorithm for decomposing convex structuring elements*, Journal of Mathematical Imaging and Vision **18** (2003), 269–289.

[9] R. Jones and I. Svalbe, *Morphological filtering as template matching*, IEEE Trans. Pattern Analysis and Machine Intelligence **16** (1994), no. 4, 438–443.

[10] A. P. Kiraly, W. E. Higgins, G. McLennan, E. A. Hoffman, and J. M. Reinhardt, *3D human airway segmentation for clinical virtual bronchoscopy*, Academic Radiology **9** (2002), no. 10, 1153–1168.

[11] D. Li and G. X. Ritter, *Decomposition of separable and symmetric convex templates*, Image Algebra and Morph. Image Proc. (San Diego, USA, July 10–12, 1990) (P. D. Gader, ed.), Vol. 1350, SPIE, Bellingham, 1990, Image Algebra and Morph. Image Proc., pp. 408–418.

[12] N. Nikopoulos and I. Pitas, *A fast implementation of 3D binary morphological transformations*, IEEE Transactions on Image Proc. **20** (2000), no. 2, 283–286.

[13] H. Park and R. T. Chin, *Decomposition of arbitrarily shaped morphological structuring elements*, IEEE Trans. Pattern Analysis and Machine Intelligence **17** (1995), no. 1, 2–15.

[14] SDC Information Systems, *SDC Morphological Toolbox for C++.* Available from: <`http://www.morph.com`> Access in: 2006-03-10.

[15] J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, London, 1982.

[16] F. Y. Shih and O. R. Mitchell, *Threshold decomposition of gray-scale morphology into binary morphology*, IEEE Trans. Pattern Analysis & Mach. Intel. **11** (1989), no. 1, 31–42.

[17] P. Soille, *Morphological Image Analysis: Principles and Applications*, 2nd ed., Springer Verlag Inc., New York, 2003.

[18] P. Soille, E. J. Breen, and R. Jones, *Recursive implementation of erosions and dilations along discrete lines at arbitrary angles*, IEEE Transactions on Pattern Analysis and Machine Intelligence **18** (1996), no. 5, 562–567.

[19] P. Soille and H. Talbot, *Directional Morphological Filtering*, IEEE Trans. Pattern Analysis and Machine Intelligence **23** (2001), no. 11, 1313–1329.

[20] M. S. Vaz and A. P. Kiraly, *An efficient method for computing mathematical morphology for medical imaging*, SPIE Medical Imaging (San Diego, USA, February 11–16, 2006) (J. M. Reinhardt and J. P. Pluim, eds.), Vol. 6144, SPIE, Bellingham, 2006, Medical Imaging 2006: Image Processing, pp. 1894–1902.

[21] X. Zhuang, *Morphological structuring function decomposition*, Computer Vision and Pattern Recognition (San Diego, USA, June 15–18, 1992), 1992, pp. 566–571.

Index

Author Index