Multi-level decomposition of Euclidean spheres

Michael Sarju Vaz[1], Atilla Peter Kiraly[2] and
Russell Manning 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 widely used in medical image analysis for applications such as segmentation and skeletonization. An efficient method for MM using convex/symmetric structuring elements (SE) is presented; this includes Euclidean discs and spheres which are of particular interest to us. Although many efficient methods for MM have been proposed in the literature, they are not readily adaptable to a true 3D Euclidean sphere SE without sacrificing accuracy for efficiency. Performance results for a chest CT dataset are presented and compared to an efficient, commercially available, software package. The results demonstrate the increasing gains of the method for larger SE: which makes the method suitable for analysis of high resolution medical images. The method is efficient for iso/anisotropic SE and, as such, allows for SE to be defined in an anatomical context (ex. 3 mm sphere) and discretized/decomposed at run-time, based on the resolution of the medical image being processed.

Keywords: mathematical morphology, structuring element, sphere, disc, circle.

1. Introduction

Mathematical morphology (MM) is based on set theory and can be used to analyze image shape features [1–3]. MM is useful for a wide variety of applications including object recognition, image segmentation, and industrial inspection [4]. In medical imaging, MM is used for such applications as brain segmentation from MR images [5] and airway/vessel segmentation from CT images [6, 7].

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 [1, 4]. 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 [6]. 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, processing time can increase polynomially with respect to SE diameter and can become prohibitive. A multitude of methods have been proposed to speed-up 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 [8, 9]: logarithmic decomposition (LD) is also very efficient for 1D and as such may be applied to separable 2D and 3D SE [3, 10]. However, only a subset of useful SE are separable. Some methods are applicable only to binary images or require preprocessing/encoding of the image [11, 12]; others cater to particular constraints such as the limited region of support of specialized hardware [13, 14].

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 [15, 16] are usually discussed and demonstrated in 2D and don't readily lend themselves to a 3D extension. So, although the sphere falls under this category, as far as we know, these methods may not be used to obtain the decomposition of the sphere. Moreover, the definition of convexity used in these methods is not consistent. Methods for decomposition of a disc SE usually sacrifice accuracy for efficiency and still do not lend themselves to the 3D sphere [3, 9, 16, 17]. Some methods refer to gray-level disk SE as spheres [3, 17]; 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 [3] may be applied to spheres, it does not readily lend itself to an efficient vectorized implementation, which is important for certain types of platforms.

SE decomposition as a union of partitions is described in [18] 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 [19]. However the decomposition is described analytically and may be hard to follow and implement. We present a reworking of this method that describes the decomposition using morphological primitives, which is much easier to follow/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 1.1 (Convex SE)**.** If the discrete SE is equal to the set of all voxels that fall inside its Euclidean convex hull [20] it is considered to be convex. Continuous domain convex shapes that are discretized satisfy this definition of digital convexity.

Definition 1.2 (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, it would be symmetric about the $x = 0, y = 0$, and $z = 0$ planes.

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, Pi, 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 Si (Equation 1b). This method will decompose any 2D or 3D convex/symmetric SE.

$$SE = P_1 \cup P_2 \cup P_3 \ldots \tag{1a}$$
$$SE = (C_1 \oplus S_1) \cup (C_2 \oplus S_2) \cup (C_3 \oplus S_3) \ldots \tag{1b}$$
$$SE = (C_1 \oplus S_1) \cup (C_1 \oplus L_2 \oplus S_2) \cup (C_1 \oplus L_2 \oplus L_3 \oplus S_3) \ldots \tag{1c}$$
$$SE = C_1 \oplus (S_1 \cup L_2 \oplus (S_2 \cup (L_3 \oplus S_3) \ldots)) \tag{1d}$$

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). 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.

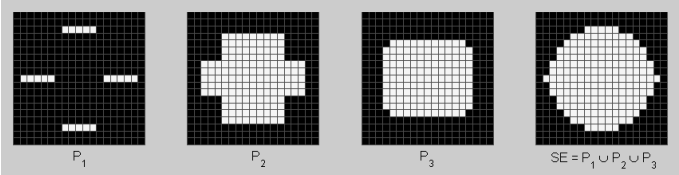

Figure 1. The proposed method decomposes the example SE (right) into a union of partitions P_1, P_2, and P_3. The origin of each partition is the center. The partitions overlap; this is possible due to the idempotency property of comparison operations (OR, AND, min, max) used to combine the partitions.

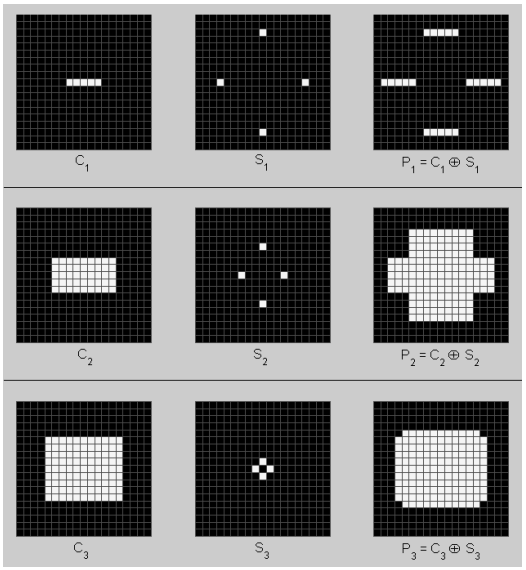

Figure 2. Each partition P_i has a cubic factor C_i and a sparse factor S_i, where $i = 1, 2, 3$ in this example. The origin of each factor is at its center.

2.2 Usage of the decomposition

Binary image dilation and erosion

$$I \oplus SE = (I \oplus P_1) \cup (I \oplus P_2) \cup (I \oplus P_3) \ldots \quad (2a)$$

$$I \oplus SE = (I \oplus C_1 \oplus S_1) \cup (I \oplus C_2 \oplus S_2) \cup (I \oplus C_3 \oplus S_3) \ldots \quad (2b)$$

$$I \oplus SE = I \oplus C_1 \oplus (S_1 \cup L_2 \oplus (S_2 \cup (L_3 \oplus S_3) \ldots)) \quad (2c)$$

Substituting $\oplus$ with $\ominus$ and $\cup$ with $\cap$ yields the equations for erosion.

Gray-level image dilation and erosion

$$I \oplus SE = \max((I \oplus P_1), (I \oplus P_2), (I \oplus P_3) \ldots) \quad (3)$$

$$I \ominus SE = \max((I \ominus P_1), (I \ominus P_2), (I \ominus P_3) \ldots) \quad (4)$$

Efficient MM for the cubic factors C_i

The cubic factors of each partition may be decomposed further. Logarithmic decomposition (LD) [3] 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 [8] 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.

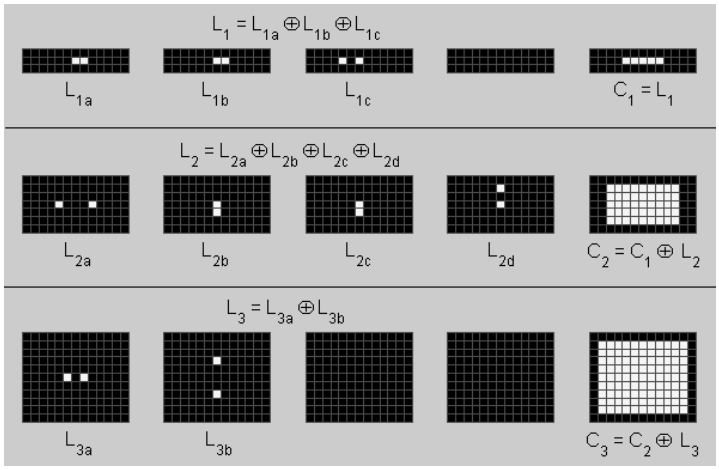

Figure 3. C_1 is a factor of C_2 and C_2 is a factor of C_3; as such, we may reuse computation across cubic factors. We use logarithmic decomposition (LD) to simply and efficiently perform the dilation/erosion of Ci. The origin of each C_i should be in its center. This can be satisfied either by assigning 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: assign SE to CSE (current SE).
2. Find the largest cube that CSE might be morphologically opened without change. This is the cubic factor C_i.

3. Find the corresponding sparse factor S_i. (Figure 5, discussion below). 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.

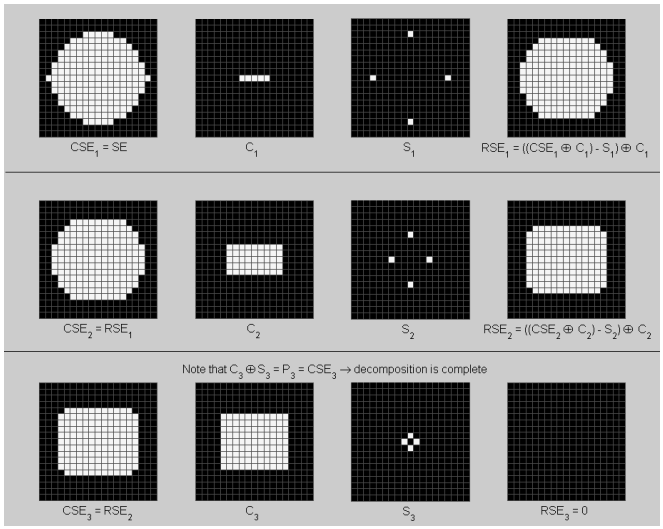

Figure 4. Decomposing the SE. At each iteration the current SE (CSE) is updated to reflect the subset of the SE that remains to be decomposed (RSE). C_i is the largest cube with which CSE_i can be morphologically opened without change. We can then determine S_i which completes the task of determining 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.
3. Select the ST_j that is sparse and has the largest number of voxels and set it to S_i.

Checking for sparseness

We enforce that our sparse factor is not face-connected for 3D or edge connected for 2D. There are a few exceptions to this rule, such as if there is connectivity with the central voxel (local origin) while in the first iteration of decomposition, i.e., during the process of determining S_1.

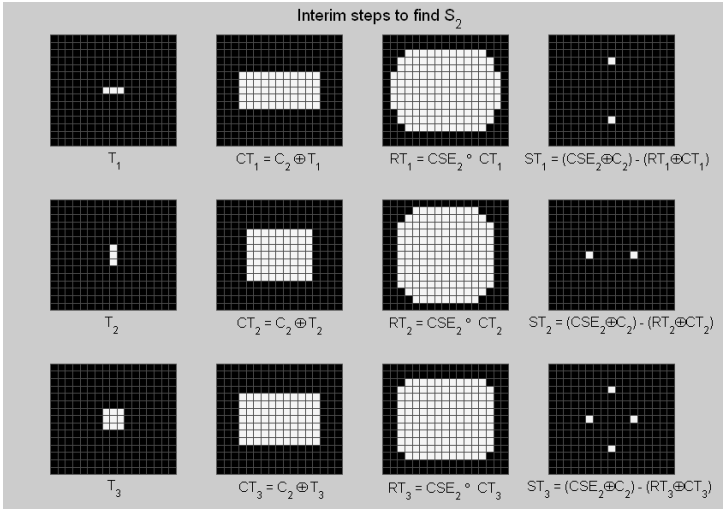

Figure 5. Determining S_2 for the considered 2D example. Obtain test cubic factors CT_j, by dilating C_i with T_j, where j = 1, 2, and 3 for 2D SE. As illustrated above, we can then obtain a set of candidate RSE, RT_j, and candidate sparse factors ST_j. Determine the subset of ST_j that is indeed sparse 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, 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 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 (slice 6 of 11) in each of the figures and is not part of S_1 and S_2, which are depicted in medium gray; S_3, however, is a single voxel at the origin. The

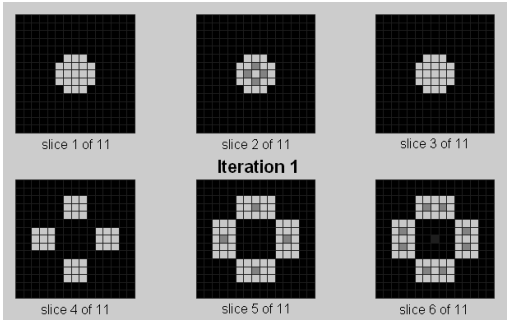

Figure 6. Upper half of P_1 for a radius-5.5 Euclidean sphere at unit quantization. The origin is the dark gray voxel in the center of slice 6, P_1 is the union of the light and medium gray voxels, medium gray - S_1. C_1 is cube-3.

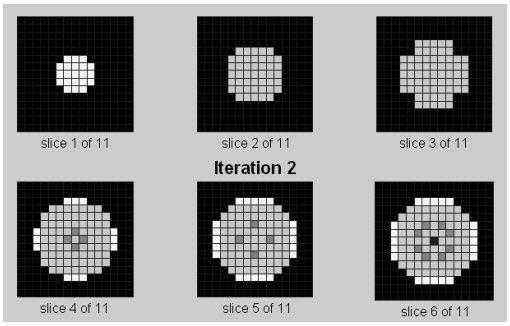

Figure 7. Foreground is $P_1 \cup P_2$ for the radius-5.5 sphere (only upper half of the sphere is shown). Dark gray - origin, light and medium gray togeather - P_2, medium gray - S_2, white - region of P_1 that does not overlap with P_2. C_2 is cube-5.

medium and light gray voxels together depict a particular partition. The gray and white voxels together in Figure 7 illustrate the 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 [21]. 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 $Xeon^{TM}$ computer with 2 GB of RAM was used. The proposed method was implemented as a single-threaded executable. As

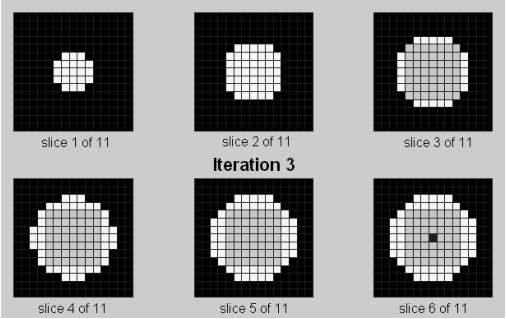

Figure 8. Dark gray - origin as well as S_3, light gray and origin together - $P_3 = C_3 =$ cube-7. White - region of $P_1 \cup P_2$ that does not overlap with P_3. The union of all foreground voxels is the union of all 3 partitions, and is the radius 5.5 Euclidean isotropic sphere.

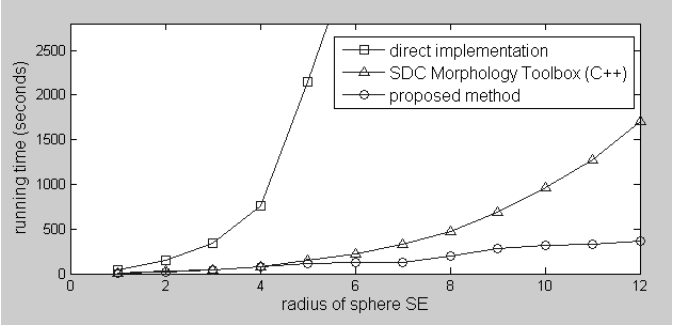

Figure 9. Processing times for gray-level dilation (binary SE) using sphere SE for a $512 \times 512 \times 418$ chest CT image. Results obtained on and Intel Pentium IV Xeon Dual CPU 2.00 GHz platform with 2.00 GB of RAM.

is evident in Figure 9, processing times for our own implementation of the 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

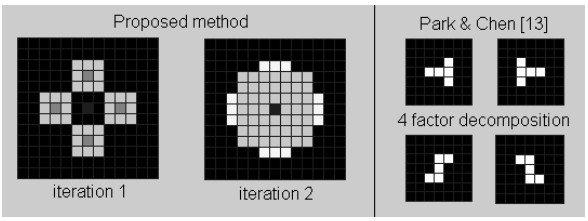

Figure 10. [Left] 2-partition decomposition of a 2D SE using the proposed method. Cubic factors are square-3 and square-7. Dark gray - origin (center): not part of S_1, however S_2 is the pixel at the origin. Light and medium gray together - the partition, medium gray (only in iteration 1) - S_1, white - 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 an iterative 2D method as would be required 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.

4. Discussion and conclusion

An efficient method for MM using disk and sphere SE is presented. 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 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.

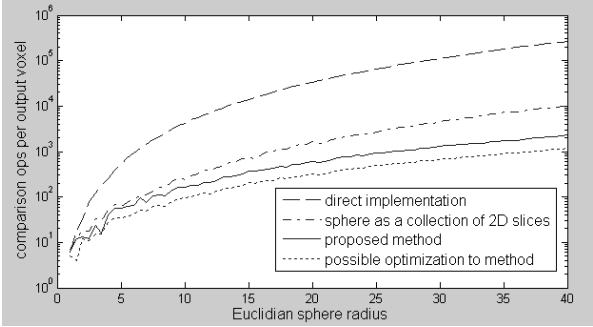

Figure 11. Number of comparison ops required per output voxel for MM using sphere SE. Note, op count presented on a log scale. Data obtained analytically. Proposed implementation follows Equation 2c and implements the sparse factors directly. A possible optimization would be to decompose the sparse factors, but this would require another interim copy of the image volume, which may be undesirable.

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 (ex: 3 mm sphere) instead of discretely, and the discrete SE could then be obtained at run-time based on the scale of the medical image under consideration: the chest CT image used to obtain the results shown in Figure 5 was quantized at a mm scale of $0.67 \times 0.67 \times 0.80$.

A comparison of proposed method with others is underway. An empirical assessment of the gains of the method in context of real medical applications will be performed. Since the method lends itself to vectorization, an even faster Graphics Processing Unit (GPU) implementation is possible.

References

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

[2] R. Gonzalez and R. Woods, *Digital Image Processing*, 1st ed., Addison-Wesley Publishing Company, Reading, 1992.

[3] V. Droogenbroeck and H. Talbot, *Fast computation of morphological operation with arbitrary structuring elements*, Pattern Recognition Letters **17** (1996), no. 14, 1451–1460.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] P. Soille and H. Talbot, *Directional Morphological Filtering*, IEEE Trans. Pattern Analysis and Machine Intelligence **23** (2001), no. 11, 1313–1329.

[10] D. Bloomberg, *Implementation Efficiency of Binary Morphology*. Available from: <http://www.leptonica.com> Access in: 2006-01-05.

[11] R. Jones and I. Svalbe, *Morphological filtering as template matching*, IEEE Trans. Pattern Analysis and Machine Intelligence **16** (1994), no. 4, 438–443.

[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] R. F. Hashimoto and J. Barrera, *A greedy algorithm for decomposing convex structuring elements*, Journal of Mathematical Imaging and Vision **18** (2003), 269–289.

[15] 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.

[16] X. Zhuang, *Morphological structuring function decomposition*, Computer Vision and Pattern Recognition (San Diego, USA, June 15–18, 1992), 1992, pp. 566–571.

[17] R. Adams, *Radial decomposition of disks and spheres*, Computer Vision, Graphics, and Image Processing **55** (1993), no. 5, 325–332.

[18] 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.

[19] 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.

[20] P. Soille, *Morphological Image Analysis: Principles and Applications*, 2nd ed., Springer Verlag Inc., New York, 2003.

[21] SDC Information Systems, *SDC Morphological Toolbox for C++*. Available from: <http://www.morph.com> Access in: 2006-03-10.