Gerenciamento de nuvem de pontos em SGBD: avaliando a extensão PointCloud para PostgreSQL

Vitor C. F. Gomes[1], Luciane Y. Sato[2], Gilberto Ribeiro de Queiroz[2], Lúbia Vinhas[2], Karine Reis Ferreira[2]

[1]Instituto de Estudos Avançados – Departamento de Ciência e Tecnologia Aeroespacial
CEP 12.228-001 – São José dos Campos – SP – Brasil

[2] Coordenação de Observação da Terra – Instituto Nacional de Pesquisas Espaciais
Caixa Postal 515 – CEP 12.227-010 – São José dos Campos – SP – Brasil

vitor@ieav.cta.br, lusato@dsr.inpe.br
{gilberto.queiroz, lubia.vinhas, karine.ferreira}@inpe.br

Abstract. *The potential observed in various applications and the advancement of technologies for point cloud data acquisition has increased the availability of this type of geospatial data. With increasing volume, new challenges arise in the efficient management of this information. Recently, the Database Management System PostgreSQL has supported this data type through the extension PointCloud. This paper explores this extension, through the analysis of its capabilities and its use in an experimental set of LiDAR data. Performance tests and disk usage metrics are collected for different compression methods. In addition, we present performance testing and disk usage metrics collected for different compression methods.*

Resumo. *O potencial observado em diversas aplicações e o avanço das tecnologias de aquisição de nuvem de pontos têm aumentado a disponibilidade desse tipo de dado geoespacial. Com o aumento do volume, novos desafios surgem quanto ao gerenciamento eficiente dessas informações. Recentemente, o Sistema Gerenciador de Banco de Dados PostgreSQL tem gerenciado esse tipo de dado através da extensão PointCloud. Este trabalho explora essa extensão, através da análise de suas capacidades e do seu uso em um conjunto experimental de dados LiDAR. Além disso, apresentamos testes de desempenho e métricas de utilização de disco coletadas para diferentes métodos de compressão.*

1. Introdução

Nuvens de pontos 3D representam uma categoria essencial de dados geoespaciais usados em uma variedade de aplicações e sistemas de geoinformação. O uso desse tipo de dado tem crescido ao longo da última década, com aplicações em diversas áreas do conhecimento, como em modelagens de objetos e construções, sítios arqueológicos, mapeamentos topográficos, aplicações florestais entre outros [Richter et al. 2015, Martinez-Rubi et al. 2016].

Tecnologias modernas de aquisição e processamento de nuvens de pontos 3D, como ecobatímetros multi-feixe, sistemas de acompanhamento de dados sísmicos e LiDAR (*Light Detection And Ranging*) em plataformas fixas ou móveis ou aeroembarcados, podem produzir de milhares a trilhões de pontos. Esses pontos, além dos dados de

posição, podem conter vários atributos, como intensidade, frequência, número de retornos, horário de coleta, entre outros [van Oosterom et al. 2015].

Dentre as tecnologias que produzem nuvens de pontos, o LiDAR tem se destacado nos últimos anos devido aos avanços na técnica de obtenção dos dados. Se por um lado essa tecnologia tem permitido a obtenção de dados com alta precisão e de grandes extensões de áreas, o armazenamento, o gerenciamento e a utilização desses dados representa um desafio devido a sua distribuição irregular, densidade e a quantidade de informações que possuem [Sabo et al. 2014].

O grande volume de dados dessas nuvens dificulta o tratamento de forma eficiente através de infraestruturas de informações e comunicação tradicionalmente utilizadas em aplicações geoespaciais. A maioria das soluções atuais de gestão desses dados apresenta limitações quanto a quantidade de dados que podem ser manipulados de forma eficiente em termos de carga de dados, armazenamento e tempo de recuperação [van Oosterom et al. 2015, Martinez-Rubi et al. 2015].

Recentemente, estruturas de dados escaláveis e estratégias para gerenciamento eficiente de nuvens de pontos têm sido exploradas com diferentes abordagens [Sabo et al. 2014, Rieg et al. 2014, Martinez-Rubi et al. 2015, van Oosterom et al. 2015, Martinez-Rubi et al. 2016]. A eficiência do sistema de gerenciamento de nuvem de pontos depende do tamanho do conjunto de dados, do *hardware* disponível e das funcionalidades requeridas pelos usuários [van Oosterom et al. 2015]. Ferramentas baseadas em processamento de arquivos, como a PDAL [Butler and Gerlek 2016] e a LAStools [Rapidlasso GmbH 2016], são amplamente utilizadas e fornecem funcionalidades para processamento em lote (*batch*) a partir da linha de comando [Martinez-Rubi et al. 2015]. Recentemente, Sistemas Gerenciadores de Banco de Dados (SGBDs), como o Oracle e PostgreSQL têm dado suporte a nuvens de pontos através de tipos específicos de dados ou extensões [Oracle 2016, Ramsey 2016].

A extensão *PointCloud* tem se destacado como uma solução para fornecer ao PostgreSQL suporte a nuvens de pontos. Resultados recentes [van Oosterom et al. 2015] mostraram bom desempenho dessa extensão em relação a solução da Oracle quando utilizando mesmo ambiente de teste.

Neste contexto, este trabalho tem como objetivo explorar as funcionalidades da extensão *PointCloud*, como as capacidades de carga de dados, seleção de pontos e seleção de estatísticas, a fim de identificar suas capacidades e avaliar seu desempenho quanto à ocupação em disco e aos tempos de carga, recuperação e consulta de dados.

2. Gerenciamento de nuvens de pontos

Em geral, há diversas possibilidades para o gerenciamento de grandes volumes de nuvens de pontos. As principais abordagens são as baseadas na manipulação de arquivos, uso de SGBDs e soluções híbridas [Rieg et al. 2014].

As soluções baseadas em arquivos armazenam os dados em arquivos ASCII ou em formatos binários, como o formato LAS [Sabo et al. 2014]. Esse formato tem se tornado um padrão de fato entre os usuários de dados LiDAR e inclui, além das coordenadas tridimensionais, atributos do retorno do laser e parâmetros de voo. Nessa abordagem, um

conjunto de ferramentas é utilizado para a seleção, análise, manipulação e visualização dos dados [van Oosterom et al. 2015]. A automatização dessas tarefas normalmente é realizada através de *scripts* para processamento em lote.

Na solução baseada em arquivos, ferramentas como a PDAL e a LAStools são frequentemente utilizadas. A PDAL (*Point Data Abstraction Library*) é uma biblioteca de licença livre com suporte a leitura e escrita dos principais formatos utilizados para nuvens de pontos, como LAS, LAZ, BPF, ILVIS2, entre outros, além de realizar a interface com servidores, como Oracle, PostgreSQL e Greyhound [Butler and Gerlek 2016]. A LAStools é um conjunto de ferramentas desenvolvidas pela *rapidlassso GmbH* e disponibilizada gratuitamente com limitações ou completa através da licença paga. Esse conjunto de ferramentas é amplamente utilizado no setor comercial e governamental [Martinez-Rubi et al. 2015, Rapidlasso GmbH 2016].

Na abordagem baseda em SGBDs, busca-se aproveitar, principalmente, os benefícios de segurança, acesso concorrente, gerenciamento de acesso, escalabilidade, indexação, particionamento, controle de versão, facilidade do uso da linguagem SQL, integração com outros formatos armazenados (vetor e raster) e integração com vasta gama de aplicações já existentes [Sabo et al. 2014]. Três métodos podem ser considerados para o armazenamento de nuvem de pontos em SGBDs: por ponto (também referenciado na literatura por *single-point* [Sabo et al. 2014] ou por *flat-model* [van Oosterom et al. 2015, Martinez-Rubi et al. 2015]), multiponto (*multipoint*) ou por bloco (também encontrado na literatura por *tile-method* [Sabo et al. 2014] ou por *block-model* [van Oosterom et al. 2015]).

No método **por ponto**, cada ponto da nuvem é armazenado em um registro no banco de dados e cada atributo representa um campo na tabela. Essa é a solução mais simples de ser implementada, pois não requer estruturas espefícias. As limitações desse método tratam-se da necessidade de indexar um grande volume de pontos, o que requer grande espaço de armazenamento, e do tempo necessário para atualizar o índice a cada atualização realizada [Sabo et al. 2014].

O método **multiponto** consiste no armazenamento de conjuntos de pontos em blocos em formato binário. Nesta abordagem, os atributos e informações geométricas são armazenados separadamente. Informações sobre a geometria do conjunto de pontos também são armazenadas para facilitar as operações espaciais sobre os dados. A limitação desse método é quanto a realização de consultas que envolvem dados espaciais e atributos, os quais estão em diferentes blocos, podendo causar problemas de desempenho. Outro fator que pode aumentar o tempo das consultas é o acesso arbitrário a pontos dentro dos blocos, os quais não são indexados [Sabo et al. 2014].

No método **por blocos**, os dados são subdivididos em pequenos blocos, onde são armazenados atributos e dados espaciais em formato binário. Neste método, também são armazenados dados sobre o envoltório do bloco de dados para auxiliar na indexação. Este método também herda as limitações da abordagem multiponto quanto ao acesso arbitrário a pontos dentro do bloco. Por conta disso, os blocos devem ter tamanho reduzido, o que gera o aumento do número de blocos e, como consequência, possíveis problemas de desempenho na indexação do grande número de registros [Sabo et al. 2014].

Na solução híbrida, SGBDs são utilizados para armazenar os envoltórios espaci-

ais dos arquivos, e um conjunto de ferramentas é utilizado para manipular diretamente arquivos de nuvem de pontos. Nesta abordagem, consultas ao banco de dados é utilizada para evitar processamento desnecessário em arquivos que não interceptam as consultas realizadas [van Oosterom et al. 2015].

As vantagens e desvantagens de cada solução são dependentes do volume de dados a serem armazenados e das operações a serem realizadas pelos usuários. Recentes trabalhos têm explorado esse tema [van Oosterom et al. 2014, Rieg et al. 2014, Sabo et al. 2014, van Oosterom et al. 2015, Martinez-Rubi et al. 2015], propondo novas estruturas de dados, algoritmos de ordenação dos dados e execução paralela de consultas.

3. Extensão PointCloud

A *PointCloud* é uma extensão para o SGBD PostgreSQL desenvolvida para lidar com o armazenamento e análise de nuvens de pontos, motivada pela necessidade de tratamento dos grandes volumes de dados gerados por sensores LiDAR. Esta extensão introduz dois novos tipos de dados: `PcPoint` e `PcPatch`. O tipo `PcPoint` é um tipo básico que representa um ponto no espaço multidimensional. O tipo `PcPatch` é uma estrutura que agrupa objetos do tipo `PcPoint` para otimizar o armazenamento dos dados. Essa estrutura permite reduzir o número de linhas necessárias para o armazenamento de uma grande quantidade de pontos. Um melhor desempenho é obtido quando os objetos do tipo `PcPoint` são agrupados por proximidade e não há sobreposição espacial entre os objetos do tipo `PcPatch` [Ramsey 2016]. Cada `PcPatch`, além de armazenar um *array* com os pontos, armazena também o retângulo envolvente dos pontos, estatísticas sobre cada dimensão e o total de pontos do *patch* [van Oosterom et al. 2015, Ramsey 2016].

Assim como a extensão PostGIS, a *PointCloud* define uma tabela e uma visão para metadados específicas do domínio de nuvens de pontos. A tabela `pointcloud_formats` é utilizada para armazenar a especificação dos atributos dos pontos que serão armazenados. A visão `pointcloud_columns` fornece a lista de todas as colunas do tipo `point_cloud`.

Muita da complexidade que existe em armazenar, gerenciar e recuperar nuvens de pontos LiDAR vem da necessidade de manipular múltiplas variáveis por ponto, as quais podem ter diferentes características, dependendo do tipo do sensor ou do modo de captura desses dados [Ramsey 2016]. Para tratar essa variabilidade, a extensão *PointCloud* utiliza um esquema XML para descrever o conteúdo dos pontos, que segue o mesmo padrão adotado pela biblioteca PDAL [Butler and Gerlek 2016]. Neste esquema, cada atributo dos pontos, chamado de dimensão, é descrito quanto a sua posição na estrutura de armazenamento, tamanho em *bytes*, tipo de dado, nome, descrição e escala. Além disso, este esquema contém informações sobre a compressão utilizada para armazenar os pontos. Atualmente a *PointCloud* permite a utilização de quatro tipos de compressão [Ramsey 2016]:

- **None**: sem nenhum tipo de compressão;
- **Dimensional**: agrupa em *arrays* cada dimensão dos pontos para compressão. Esse modo é indicado para grupos de dados homogêneos;
- **GHT** (**GeoHashTree**): armazena os pontos em uma árvore onde cada nó armazena os valores comuns compartilhados por todos os nós descendentes [Sabo et al. 2014]. Esse modo de compressão é indicado para grupos com grande quantidade de pontos;

- **LAZ**: utiliza o sistema de compressão LASZip, que é um sistema de compressão aberto e gratuito desenvolvido pela *rapidlasso GmbH* [Rapidlasso GmbH 2016].

Atualmente, a *PointCloud* fornece 28 funções para a manipulação e extração de estatísticas sobre os tipos `PcPoint` e `PcPatch`. Outra característica importante da *PointCloud* é sua integração com a extensão PostGIS, desenvolvida com uma extensão própria chamada *PointCloud PostGIS*. Através dessa integração é possível, por exemplo, realizar consultas para obter o conjunto de *patches* que interceptam uma dada geometria (`PC_Intersection(pcpatch, geometry)`) ou verificar se uma dada geometria intercepta um *patch* (`PC_Intersects(pcpatch, geometry)`).

4. Avaliação da Extensão PointCloud

A fim de avaliar as capacidades da extensão *PointCloud* e seu desempenho quanto aos diferentes modos de compressão frente a tarefas como a carga e recuperação de dados e consultas sobre uma nuvem de pontos, dois conjuntos de testes foram realizados. No primeiro, foi avaliado o espaço ocupado em disco e tempo necessário para a carga e recuperação de uma nuvem de pontos. Neste teste, foram obtidos os tempos médios de carga e recuperação para as quatro opções de compressão disponíveis na extensão *PointCloud*. Para o segundo conjundo de testes, foram avaliados os tempos necessários para a execução de consultas frequentemente utilizadas por usuários de dados desta natureza [Suijker et al. 2014, van Oosterom et al. 2015]. As consultas escolhidas para o teste foram:

- Seleção de pontos que interceptam um polígono retangular (C1);
- Seleção de pontos que interceptam um polígono irregular (C2);
- Seleção de atributo (C3);
- Seleção de estatísticas por *patch* (C4); e
- Seleção de estatísticas globais (C5).

As consultas em SQL de cada seleção são apresentadas na Listagem 1.

Listing 1. Consultas utilizadas nos testes de seleção de pontos

```
-- Selecao de pontos que interceptam um poligono retangular (C1)
CREATE TABLE result1 AS(
  SELECT 1 as id,
    PC_Union(PC_Intersection(l.pa, r.geom)) AS pa
    FROM lidar_tlb l, recorte r WHERE r.gid = 1);

-- Selecao de pontos que interceptam um poligono irregular (C2)
CREATE TABLE result2 AS(
  SELECT 1 as id,
    PC_Union(PC_Intersection(l.pa, r.geom)) AS pa
    FROM lidar_tlb l, recorte r WHERE r.gid = 2);

-- Selecao de atributos (C3)
SELECT PC_Get(PC_Explode(pa), 'Z') FROM lidar_tlb;

-- Extracao de estatisticas por \textit{patch} (C4)
SELECT PC_Summary(pa) FROM lidar_tlb;
```

```
-- Extracao de estatisticas globais (C5)
SELECT PC_Summary(PC_Union(pa)) FROM lidar_tlb WHERE id <= 20000;
```

Para a realização dos testes, foram utilizados dois arquivos LAS com dados coletados por um instrumento LiDAR modelo *Optec Orion*. O primeiro arquivo (Arquivo 1) ocupa 79 MB, possui 2.933.330 pontos, foi adquirido sobre uma área de 46.168 m^2 e possui em média 63,5 pontos por metro quadrado. O segundo arquivo (Arquivo 2) utilizado para os testes possui 1,8 GB, 65.452.394 pontos, representa uma área sobrevoada de 850.291 m^2 e possui em média 76,98 pontos por metro quadrado. Os arquivos possuem 13 dimensões, incluindo as dimensões X, Y e Z, e utilizam o sistema de referência SIRGAS 2000 / UTM zona 20S (`EPSG:31980`).

Para cada arquivo, foi criada uma base de dados com cada um dos 4 tipos de compressão disponíveis na extensão *PointCloud*. Em cada base de dados foi criada uma tabela com duas colunas. Um identificador (id) sequencial do tipo *integer* e um campo para o armazenamento dos `PcPatch` (pa).

A plataforma utilizada para os testes consiste de um servidor com processador i7-5820K 3.30 GHz (6 cores, 2 threads por core), 15MB Cache e 16 GB de memória RAM. Para o armazenamento dos dados, utilizou-se um disco SSD. O sistema operacional utilizado foi o Ubuntu 14.04 64-bit. A extensão *PointCloud* foi instalada em um servidor PostgreSQL versão 9.5. Para a carga dos dados, foi utilizada a biblioteca PDAL versão 1.2.0 [Butler and Gerlek 2016].

Para ambos os arquivos criados, foram utilizados *patches* com até 400 pontos, criados através do filtro `filters.chipper` da PDAL. Para a carga dos dados utilizando o modo ght, foi necessário converter os dados para um sistema com coordenadas geográficas, pois o algoritmo somente aceita coordenadas entre os valores -180/180 e -90/90. Para isso, foi utilizado o filtro `filters.reprojection` da PDAL para converter do sistema de referência SIRGAS 2000 para o WGS 84 (`EPSG:4326`). Para todas as cargas, foi utilizado o esquema XML gerado pela PDAL e gravado automaticamente na tabela `pointcloud_formats` durante a importação. A exceção foi com o modo de compressão LAZ, que por padrão gerou um esquema para o modo sem compressão, diferente do informado no arquivo de configuração. Para garantir o uso do modo de compressão LAZ, o esquema foi criado manualmente e informado à PDAL durante a carga dos dados.

A partir da definição dos testes, foram elaborados *scripts* e consultas SQLs para serem executados em cada uma das base de dados. Para a coleta do tempo de carga e recuperação dos dados foi utilizada a ferramenta `time` do terminal do Linux. Para as consultas em SQL, foi utilizado o comando `\timing` do `psql`. O tamanho ocupado por cada base de dados foi obtido através do comando `\l+` do `psql`.

Para o segundo conjunto de testes, cada consulta foi executada 6 vezes, sendo descartado o tempo da primeira execução, e calculado o tempo médio das 5 restantes. Esse procedimento foi adotado para evitar que dados em *cache* afetassem seletivamente algumas das consultas em análise. Antes da realização das consultas, a base foi otimizada através do comando `VACUUM FULL`.

5. Resultados

Os resultados quanto ao espaço ocupado em disco e tempo de carga e recuperação dos dados dos dois arquivos são apresentados na Tabela 1. Os valores ausentes na tabela indicam erros ocorridos na extensão *PointCloud* durante a execução dos testes. Para ambos os conjuntos de testes ocorreram erros associados aos limites no tamanho da consulta retornada, os quais são específicos para cada modo de compressão.

Com os resultados, é possível observar que há pouca variação na velocidade de carga e recuperação quando comparamos os diferentes modos de compressão. As versões sem compressão (none) e LAZ possuem velocidades equivalentes e melhores em relação as demais (dimensional e ght) para ambos os tamanhos de arquivos. Cabe destaque a variação de espaço necessário para armazenar os dados nos diferentes modos de compressão. O modo de compressão LAZ foi o que apresentou menor tamanho final da base de dados, seguido pelo dimensional. Como esperado, o método sem compressão foi o modo que requisitou mais espaço em disco para armazenar os pontos.

Tabela 1. Tempos médios dos testes de carga e recuperação de dados

	Arquivo 1 (79 MB)				Arquivo 2 (1,8 GB)			
Atributo / Compressão	none	dim.	ght	LAZ	none	dim.	ght	LAZ
Tamanho (MB)	157	51	138	20	3257	886	2630	158
Tempo carga (s)	45,83	49,39	53,89	45,98	1015,23	1090,40	1328,58	1034,31
Vel. carga (MB/s)	1,72	1,60	1,47	1,72	1,81	1,68	1,38	1,81
Tempo recuperação. (s)	29,90	29,76	35,42	27,81	651,02	647,35	853,98	-
Vel. recuperação (MB/s)	3,21	3,23	3,71	3,45	3,42	3,44	2,61	-

ght: GeoHashTree; dim: dimensional

Os tempos das consultas realizadas são apresentados na Tabela 2. Para a consulta C5 aplicada às tabelas com dados do Arquivo 2, foi necessário restringir a seleção em 20.000 *patches*, devido a limitação de memória a ser utilizada pelo retorno da consulta. Atualmente existe uma *Issue* (#46) aberta que trata desse problema no repositório da extensão *PointCloud*.

Tabela 2. Tempos médios das consultas

	Arquivo 1 (79 MB)				Arquivo 2 (1,8 GB)			
Consulta / Compressão	none	dim.	ght	LAZ	none	dim.	ght	LAZ
C1 (s)	6,52	11,34	12,73	6,34	113,64	132,47	-	138,73
C2 (s)	6,19	9,22	12,70	6,36	114,46	132,39	-	138,72
C3 (s)	4,39	5,16	11,23	8,68	95,46	113,06	249,82	209,05
C4 (s)	0,264	0,290	0,258	0,237	4,41	4,79	4,25	3,44
C5 (s)	0,713	22,11	20,68	2,18	1,64*	57,10*	57,07*	5,79*

ght: GeoHashTree; dim: dimensional;
*: consulta limitada aos 20.000 *patches*.

De maneira geral, observa-se que sem compressão as consultas são realizadas em menor tempo. Este comportamento era esperado, uma vez que no modo sem compressão, os dados podem ser acessados sem a necessidade de serem descompactados. Para o Arquivo 1, outro modo de compressão que se destaca é o LAZ, sendo o segundo melhor tempo em quase todas as consultas. O desempenho observado pode ser justificado pelo fato do modo de compressão LAZ permitir que os dados compactados possam ser acessados diretamente pela aplicação, sem a necessidade de serem descompactados previamente

em disco [Isenburg 2011]. A exceção é observada na consulta C3, onde o modo de compressão dimensional tem o segundo melhor tempo de resposta. A consulta C3 trata da seleção de um atributo específico dos pontos, o que pode justificar o melhor desempenho, uma vez que neste modo de compressão as dimensões são armazenadas separadamente.

Ainda para o Arquivo 1, observa-se um pobre desempenho para a operação da extração de estatística globais para os modos de compressão dimensional e ght. Esse mesmo comportamento é visto para o Arquivo 2. Nesta consulta, todos os *patches* são agrupados em um único `PcPatch` para, em seguida, ser realizado o cálculo das estatísticas.

Para o Arquivo 2, destaca-se o modo de compressão dimensional para as consultas C1, C2 e C3. Para as consultas C4 e C5, o modo LAZ teve melhor desempenho.

6. Considerações Finais

Este trabalho apresenta o estado da arte das ferramentas de software livre existentes para o gerenciamento de nuvens de pontos. Além disso, faz uma avaliação da extensão *PointCloud* através de testes em 4 modos de compressão para avaliar o seu funcionamento, o uso de espaço em disco e o tempo necessário para a realização de consultas frequentemente utilizadas.

Os resultados obtidos indicam que os modos de compressão afetam significativamente o espaço ocupado pelos dados em um SGBD e o tempo necessário para a realização das consultas. Verifica-se que de fato a decisão sobre qual modo utilizar depende do volume dos dados a serem geridos e das operações que serão realizadas sobre eles, como também é observado no trabalho de [van Oosterom et al. 2015].

As limitações e erros encontrados com o uso dos métodos de compressão ght e LAZ indicam ainda baixa maturidade dessas implementações na extensão *PointCloud*. Pelos testes realizados neste trabalho, acreditamos que o modo de compressão *dimensional* ou sem compressão sejam os mais adequados para uso no momento.

Uma das vantagens da extensão *PointCloud* é ter disponível uma biblioteca como a PDAL e as ferramentas associadas para realizar a carga e recuperação dos dados.

Os resultados obtidos neste trabalho nos motivam a estabelecer uma continuidade. Planejamos avaliar um grupo mais amplo de funções e tamanhos diferentes de *patches* em um conjunto maior de dados LiDAR, compatível com levantamentos que estão sendo realizados por projetos como o Paisagens Sustentáveis Brasil - EMBRAPA/USFS e o Monitoramento Ambiental por Satélite no Bioma Amazônia - INPE. Além disso, planeja-se estruturar e realizar experimentos que comparem a utilização de nuvens de pontos utilizando a abordagem por arquivos e a baseada em SGBD em uma aplicação real dos dados.

Agradecimentos

Os dados LiDAR utilizados neste trabalho foram adquiridos pelo Projeto Paisagens Sustentáveis Brasil, apoiado no Brasil pela Embrapa e pelo Serviço Florestal, Agência para o Desenvolvimento Internacional e Departamento de Estado dos Estados Unidos.

Referências

Butler, H. and Gerlek, M. (2016). PDAL - Point Data Abstraction Library. http://www.pdal.io. Acessed: 04 sep 2016.

Isenburg, M. (2011). Laszip: lossless compression of lidar data. http://lastools.org/download/laszip.pdf. Acessed: 29 oct 2016.

Martinez-Rubi, O., de Kleijn, M., Verhoeven, S., Drost, N., Attema, J., van Meersbergen, M., van Nieuwpoort, R., de Hond, R., Dias, E., and Svetachov, P. (2016). Using modular 3d digital earth applications based on point clouds for the study of complex sites. *International Journal of Digital Earth*, pages 1–18.

Martinez-Rubi, O., van Oosterom, P., Gonçalves, R., Tijssen, T., Ivanova, M., Kersten, M. L., and Alvanaki, F. (2015). Benchmarking and improving point cloud data management in monetdb. *SIGSPATIAL Special*, 6(2):11–18.

Oracle (2016). Point cloud-related object types. http://docs.oracle.com/database/121/SPATL/GUID-D347DC78-8782-4BB5-995D-0315B3DD2AB4.htm. Acessed: 04 sep 2016.

Ramsey, P. (2016). Pointcloud: a PostgreSQL extension for storing point cloud (LIDAR) data. https://github.com/pgpointcloud/pointcloud. Acessed: 04 sep 2016.

Rapidlasso GmbH (2016). Lastools. https://rapidlasso.com. Acessed: 04 sep 2016.

Richter, R., Discher, S., and Döllner, J. (2015). Out-of-core visualization of classified 3d point clouds. In *Lecture Notes in Geoinformation and Cartography*, The Selected Papers of the 3D GeoInfo 2014, pages 1863–2246. Springer International Publishing.

Rieg, L., Wichmann, V., Rutzinger, M., Sailer, R., Geist, T., and Stötter, J. (2014). Data infrastructure for multitemporal airborne lidar point cloud analysis – examples from physical geography in high mountain environments. *Computers, Environment and Urban Systems*, 45:137 – 146.

Sabo, N., Beaulieu, A., Belanger, D., Belzile, Y., and B., P. (2014). The geohashtree: a multi-resolution data structure for the management of point clouds. Technical notes 4, Minister of Natural Resources Canada.

Suijker, P., Alkemade, I., Kodde, M. P., and Nonhebel, A. (2014). User requirements massive point clouds for esciences (wp1). Technical report, DelftUniversityof Technology.

van Oosterom, P., Martinez-Rubi, O., Ivanova, M., Horhammer, M., Geringer, D., Ravada, S., Tijssen, T., Kodde, M., and Gonçalves, R. (2015). Massive point cloud data management: Design, implementation and execution of a point cloud benchmark. *Computers & Graphics*, 49:92 – 125.

van Oosterom, P., Ravada, S., Horhammer, M., Rubi, O. M., Ivanova, M., Kodde, M., and Tijssen, T. (2014). Point cloud data management. *IQmulus Workshop on Processing Large Geospatial*.