SEGMENTATION OF LIDAR POINT CLOUDS FOR BUILDING EXTRACTION

Jun Wang **Jie Shan**
{wang31, jshan}@ecn.purdue.edu
Geomatics Engineering, School of Civil Engineering, Purdue University
West Lafayette, IN 47907, USA

ABSTRACT

The objective of segmentation on point clouds is to spatially group points with similar properties into homogeneous regions. Segmentation is a fundamental issue in processing point clouds data acquired by LiDAR and the quality of segmentation largely determines the success of information retrieval. Unlike the image or TIN model, the point clouds do not explicitly represent topology information. As a result, most existing segmentation methods for image and TIN have encountered two difficulties. First, converting data from irregular 3-D point clouds to other models usually leads to information loss; this is particularly a serious drawback for range image based algorithms. Second, the high computation cost of converting a large volume of point data is a considerable problem for any large scale LiDAR application. In this paper, we investigate the strategy to develop LiDAR segmentation methods directly based on point clouds data model. We first discuss several potential local similarity measures based on discrete computation geometry and machine learning. A prototype algorithm supported by fast nearest neighborhood search and based on advanced similarity measures is proposed and implemented to segment point clouds directly. Our experiments show that the proposed method is efficient and robust comparing with algorithms based on image and TIN. The paper will review popular segmentation methods in related disciplines and present the segmentation results of diverse buildings with different levels of difficulty.

INTRODUCTION

LiDAR collects high resolution Earth surface information as the scattered and unorganized dense point clouds. Airborne laser scanning (ALS) systems are well suited for the building detection and reconstruction in large urban scenes. Terrestrial LiDAR systems are able to capture building facade details in close range. The building detection are essentially a segmentation process, which separates points on buildings from the points on the surfaces of other landscape contents, such as ground, trees and roads. We adopt the formal definition of range image segmentation as given in Hoover et al. (1996). The segmentation of point clouds can be defined as the following:

Let *R* represent the spatial region of the entire point clouds *P*. The goal of the segmentation is to partition *R* into sub-regions R_i, such that:

1. $\bigcup_i^n R_i = R$
2. $R_i(P_i) \neq \phi$ for any i, $\bigcup_i^n R_i(P_i) = P$
3. R_i is a connected region, $i = 1, 2, \ldots n$
4. $R_i \cap R_j = \emptyset$, for all i and j, $i \neq j$
5. $L(R_i)$ = True for $i = 1, 2, .., n$ and
6. $L(R_i \cup R_j)$ = False for $i \neq j$

where $L(R_i)$ is a logical predicate over the points P_i in the region R_i, which defines the measure of similarity that groups the points in one region and separates them from the points in other regions.

Segmentation is one of the most active LiDAR research areas. There are many research activities and real-world applications dealing with segmentation. Without loss of generality, we summarize the existing schemes in the following ways.

Based on the different outputs of the algorithms, segmentation methods might be classified in two types: part-type segmentation and patch-type segmentation. Part-type methods try to segment LiDAR data into visually meaningful simple objects by directly extracting primitive geometric feature such as planes, cylinders and spheres. All these simple objects can be described with a few mathematical parameters. Hough transform (Vosselman et al., 2004), RANSAC (RANdom SAmple Consensus) (Schnabel et al., 2004) and least square fitting (Ahn, 2004) are some common techniques used in part-type segmentation algorithms. Part-type algorithm mainly works better with application using terrestrial LiDAR data, such as extracting building parts and reverse engineering of industrial site. Patch-type methods segment point clouds to homogeneous regions based on proximity of points or similarity of locally estimated parameters. Later on, these surface patches are further classified and organized into more meaningful structures. Most algorithms dealing with airborne LiDAR fall into this category.

Based on the different ways to represent homogeneous regions, segmentation methods generally fall into two classes: edge/boundary based and surface based. Edge based methods use a variety of methods to outline boundaries or detect edges of different regions with respects to local geometry properties, then group the points inside the boundaries to give final segmentations. As for the surface based methods, local geometry properties are used as a similarity measure during segmentation. The points that are spatially close and have similar surface properties are merged together to form different regions, from which boundaries are then deduced.

Based on mathematical techniques used, the published segmentation algorithms can be roughly categorized into five groups:

1. Edge-detection method: Heath et al. (1998), Jiang and Bunke (1999), Sappa and Devy (2001). A large variety of edge-detection algorithms have been developed for image segmentation in computer vision area (Shapiro and Stockman 2001). LiDAR data are converted into range image, e.g. DSM to make it suitable to image edge-detection methods. The performance of segmentation is largely dependent on the edge detector. However, the operation of converting 3D point clouds to 2.5D range images inevitably causes information loss. For airborne LiDAR data, the overlapping surface such as multi-layer building roofs, bridges, and tree branches on top of roofs cause buildings and bridges either under segmented or wrongly classified. The point clouds obtained by terrestrial LiDAR are usually combined from the scans in several different positions, converting such kind of true 3D data into 2.5D would cause great loss of information.
2. Surface-growing method: Gorte (2002), Lee and Schenk (2002), Rottensteiner (2003), Pu and Vosselman (2006), Rabbani et al. (2006). Surface growing in point clouds is comparable to region growing in images. First, seeds, which can be planar or non-planar surface patches, are indentified. Least squares adjustment and Hough transform are robust methods to detect planar seeds. The seeds are then extended gradually to larger surface patches by grouping points around them based on similarity measures, such as proximity, slope, curvature and surface normal. Surface-growing algorithms are widely used in LiDAR segmentation, since they are easy to implement and computational cost is relatively low for large point clouds. However, seeds selection can be a problem for region-growing methods. It is difficult to judge if one set of seeds selection is better than the other set. Also different set of seeds can lead to different segmentation results. Surface-growing methods are efficient in their own ways, however, generally not regarded as robust methods.
3. Scan-line methods: Jiang and Bunke (1994), Sithole and Vosselman (2003), Khalifa et al. (2003). Scan-line methods adopt a split-and-merge strategy. Range image splits into scan lines along a given direction, for example, each row can be considered as a scan line. For a planar surface, a scan line on any 3D plane makes a 3D straight line. Each scan line is segmented independently into line segments until the perpendicular distance of points to their corresponding line segment is below certain threshold. Then segments from scan lines are merged together based on some similarity measures in a region-growing fashion. The scan line method is based on 2.5D grid model and mainly designed to extract planar surfaces. Scan lines do not exist in unstructured point clouds. Extension of scan-line methods into point clouds requires deciding the preferred directions and constructing scan lines by slicing point clouds, this makes segmentation results depend on orientation.
4. Clustering methods: Roggero (2001), Filin (2002), Biosca and Lerma (2008), Chehata et al. (2008). In these algorithms, each point is associated with a feature vector which consists of several geometric or radiometric measures. LiDAR points are then segmented in feature space by clustering technique, such as k-means, maximum Likelihood and fuzzy-clustering; unlike the other methods, clustering is carried out in the feature space, it can works on point clouds, grid and TIN. Clustering methods have shown their ability to perform robust segmentation on both airborne and terrestrial laser scanner point clouds. The performance of the clustering algorithms is decided by the choice of feature vectors and clustering technique.
5. Graph partitioning methods: Kim and Muller (1999), Fuchs (2001), Wang and Chu (2008). Points in the same segment are much more closely connected to each other than they are to points in other segment. So the boundary between two segments must lie on the place that has the weakest connection. This simple idea is captured by constructing proximity graphs or neighborhood graphs. A proximity graph is an attribute graph $G(V, E)$ on the point clouds. Each point is a node in the set V and the set E consists of the connections/edges between a pair of points. Each edge has a weight to measure the similarity of the pair of points. The segmentations is achieved by graph partitioning algorithms to find the optimized cuts that minimize the similarity between segments while maximizing the similarity within segments at the same time. Segmentation can be performed as recursive partitioning or direct multi-way partitioning. Normalized cut (Shi and Malik,

2000), Minimum spanning tree (MST) (Haxhimusa and Kropatsch, 2003) and spectral graph partitioning (Chung, 1997) are widely used graph decimation algorithms.

The results of segmentation process are homogeneous regions with respect to some similarity measures. However, it doesn't always lead to a final meaningful segmentation in term of semantics. Evaluation of the segmentation algorithms depends on many factors: the properties of algorithms, such as complexity, efficiency and memory requirements; the parameters of the algorithm; the type of data: real world data or synthetic data; the type of LiDAR: airborne or terrestrial; the landscape setting of LiDAR data: urban or rural, steep or flat area; the method used for evaluation. It is very difficult to develop an evaluation method to find the optimal segmentation algorithm, since segmentation is often application specific. A meaningful segmentation can be quite different in different applications and landscape settings. The user intervention shall play an important role to evaluate the segmentation algorithms.

It is noticeable that most of the existing segmentation methods are based on 2.5D range image or TIN model. There are many limitations on their ability to extend algorithms to 3D unstructured point clouds. On the other hand, converting data from one model to another model usually leads to information loss; this is particularly a serious drawback for range image based algorithms. Also the high computing cost of model converting is a considerable problem for any large-scale LiDAR application. LiDAR data requires segmentation on point clouds directly.

In point clouds generated by airborne LiDAR system, the structure of a building generally can be described by two types of edges: jump edge and crease edge. Jump edge is defined as discontinuities in depth or height values. Such edges separate one building from another building and other ground objects. Creases edge is formed where two surfaces meet. Such edges are characterized by discontinuities in surface normals, and the surface normals at the intersection line make an angle greater than the given threshold. These two types of edges serve different purposes in segmentation. Generally speaking, jump edges aim for object extraction, and crease edges are used more often in object reconstruction. Jump edges are used to segment point clouds into the surface patches from different objects. In post-processing stage, the surface patches are grouped into identifiable objects, such as ground, buildings and trees. Crease edges are used to further segment points into adjacent planar regions, which are inside the surface patch defined by jump edges. The roof structure of a building is a good example of crease edges.

In this research, we present an approach to extract buildings from airborne LiDAR data by directly discovering 3D jump edges in point clouds.

BUILDING EXRACTRATION ALGORITHM

The process of building extraction is separated into three steps: first, discovering points on jump edges by the nearest neighborhood based computation which minimizes the memory requirement for large points set. Second, connecting points to form jump edges to approximate the building outlines. And in third step, buildings and trees are separated.

Step 1: labeling jump edges points

Jump edges defined the building outlines in airborne LiDAR point clouds; it reflects the shape of points on top of building. The convex hull can quickly captures a rough idea of the shape or extent of a point data set with the relatively low computing cost. Figure 1 shows an example to capture the shape of the random generated point set. The global convex hull fails to capture the concave part both in 2-dimensional and 3-dimensional convex hull. However, it doesn't mean convex hull is not suitable for our purpose; it has interesting properties for further development.

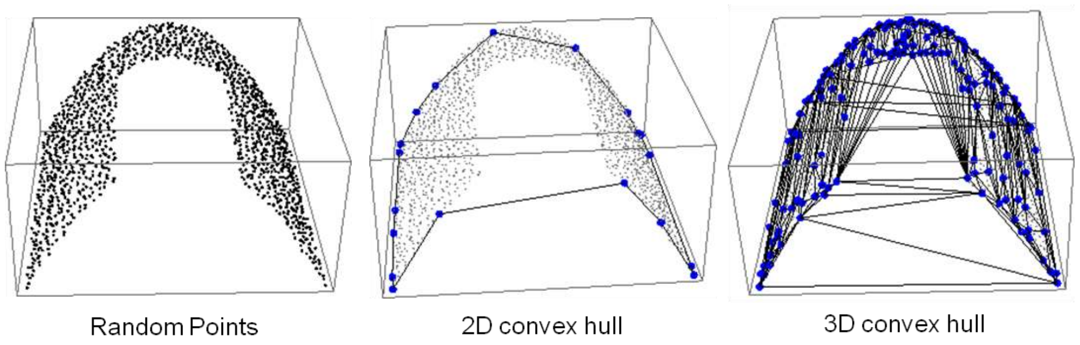


Figure 1 Random generated point set and the global convex hulls

Convex hull can capture the rough shape of the point set and classify the points into two groups: boundary points and non-boundary/inside points. Non-boundary points can indentified without constructing convex hull.

Theorem 1. A points p fails to be an extreme point (boundary point) of a plane convex set S only if it lies in some triangle whose vertices are in S but is not itself a vertex of the triangle (see **Figure 2**). (Preparata and Shamos 1985). In 3-dimensional space, the tetrahedron replaces the triangle. And n-dimensional space, n-simplex is an n-dimensional analogue of a triangle in 2-dimensional space.

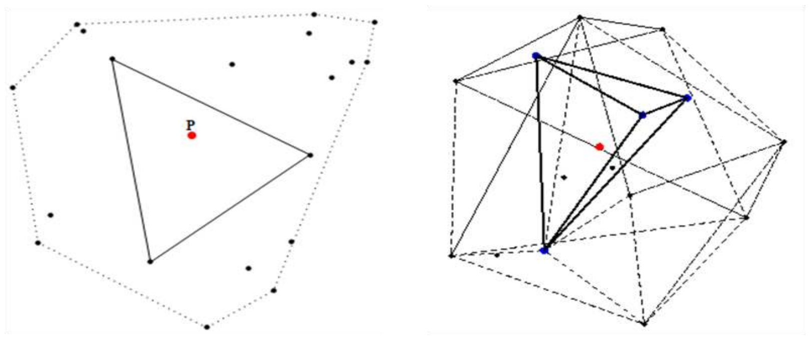


Figure 2 Example of Theorem 1

Theorem 2. For a convex set S, pick up any subset of points contain point p, if p is inside the convex hull of the subset of points, then p is not on the boundary of the convex hull of points S.

The above Theorems produce a simple way to test if a point is non-boundary point based on convex hull constructed by the point and its nearest neighbors. The complex boundary of the points set can be approximated by removing non-boundary points by local convex hull testing. This process can be viewed as using much smaller size local convex hulls sculpture out the shape of the whole points set. Algorithm 1 (Figure 3) is designed based on this simple idea.

```
Algorithm 1. Basic boundary labeling algorithm
Input (points, n)
Output: { boundary points}
1.      For all the points:
2.          If point p is not labeled as "non-boundary point"
3.              Pick up n nearest neighbors of point p
4.              Construct convex hull of (p, {$p_i$})
5.              Label all the points insides the convex hull as "non-boundary" points
6.      Label all the non-labeled points as "boundary point"
```

Figure 3 Basic boundary labeling algorithm

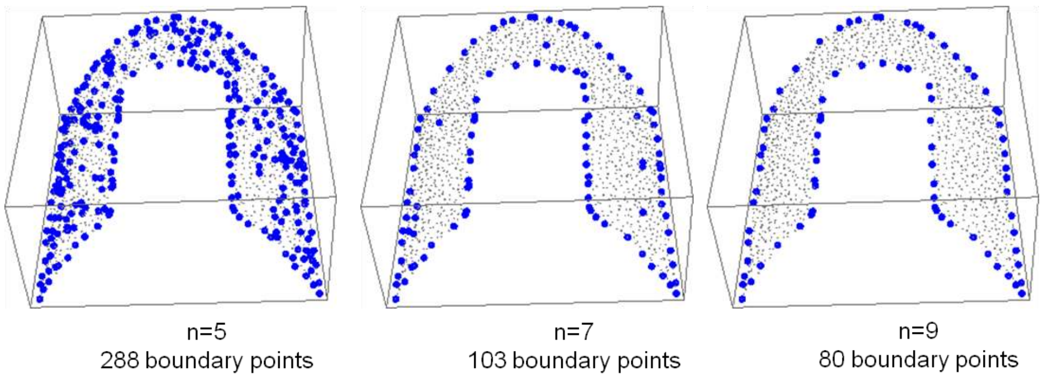


Figure 4 Examples of Algorithm 1

The only parameter is n, the number of nearest neighbors. Figure 4 shows the results with different values of parameter n. When n is increased to 9, all the inside points have been eliminated, the result correctly capture the non-convex part of the boundary. The higher n goes, the less points picked up as boundary points, the general range of n (9 to 15) is recommended from the balance of effectiveness and efficiency.

Algorithm 1 is enhanced by adding another parameter. In Algorithm 2 (Figure 5), the distance parameter ε is added to pick up the points which are close enough to the local convex boundary and can be treated as "boundary points". The higher ε goes, more points picked up on boundary. However, when ε is above the certain value, the nonboundary points are also been marked as "boundary point" (**Figure 6**). The general guide line is that ε shall be less than the average distance between any two nearest pair. A more realistic example shows in Figure 7. Boundary points on jump edges are identified and also the points on the corresponding boundary on the grounds.

```
Algorithm 2. Enhanced boundary labeling algorithm
Input (points, n, ε)
Output: { boundary points}
1.   For all the points:
2.       Pick up n nearest neighbors of point p
3.       Construct convex hull of (p, {p_i})
4.       Calculate the minimum distance from point p to the convex hull boundary
5.       If dist > ε, label point p as "non-boundary point"
6.   Label all the non-labeled points as "boundary point"
```

Figure 5 Enhanced boundary labeling algorithm

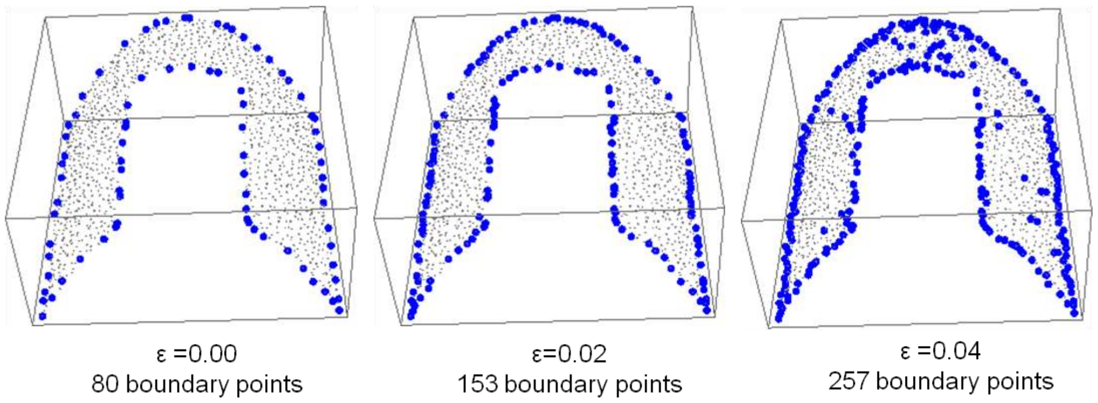


Figure 6 Examples of Algorithm 2

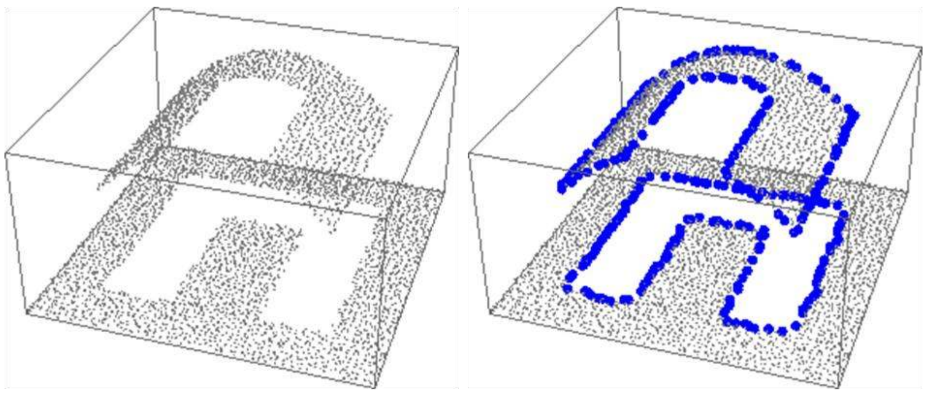


Figure 7 A more realistic example

Step 2: connecting points to form jump edges

In this step, boundary points need to be grouped and connected to lines which represent the jump edges. We present an algorithm based on k-nearest-neighbor network and minimum spanning trees (MST). K-nearest-neighbor network is a weight graph G(V, E), where the vertices of graph (V) are the boundaries points, and edges (E) are the connections between the point and its k nearest neighbors, the weight is the distance between two connected points.

For each edge, if its weight is large than the certain value ε, this edge is discarded. This is used to get rid of isolated points. For each sub-graphs, the jump boundaries is formed by MST. MST is a tree that passed though all the vertices of a given graph with minimum total weight.

```
Algorithm 3. Points connecting algorithms
Input (boundary points, k, ε)
Output: { jump edges}
1.      For all the points:
2.          Pick up k nearest neighbors of point p
3.          Construct graph edges from point p to its nearest neighbors, distance as the weight
4.      Discard edges whose weight is larger than ε
5.      Partition knn-graph into disconnected sub-graphs if necessary
6.      Calculate MST for each sub graph
7.      Return MSTs as jump edges
```

Figure 8 Points connecting algorithm

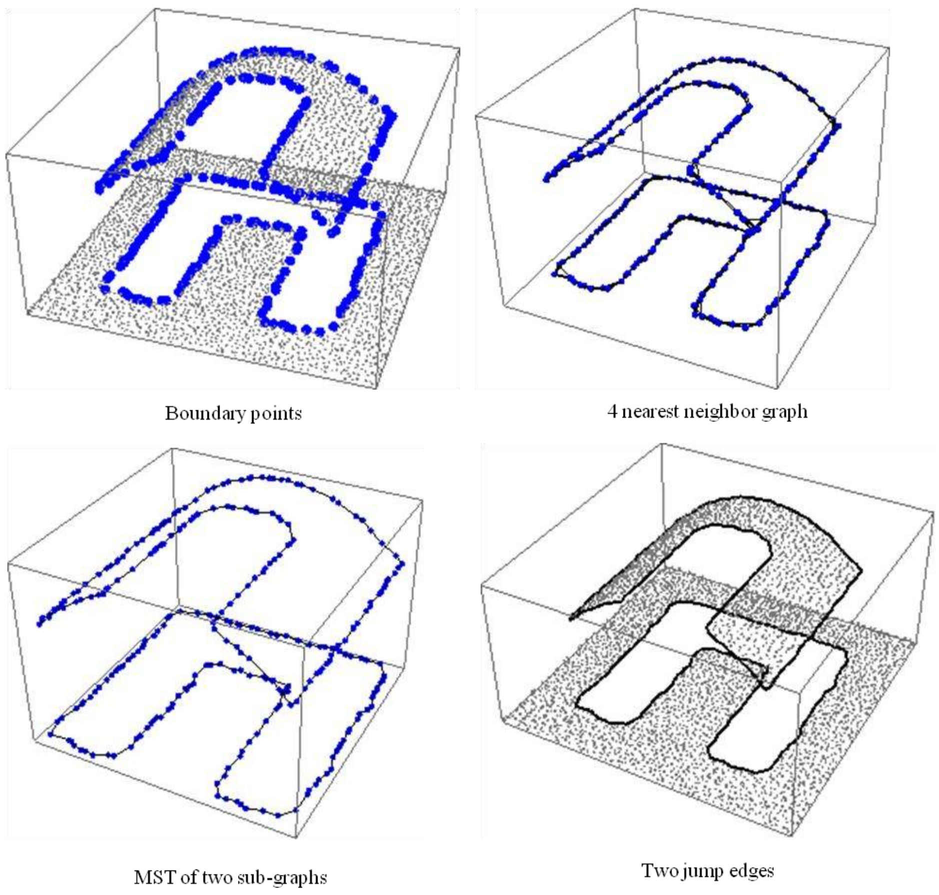


Figure 9 Example of algorithm 3

Step 3: separate building and trees

Separating points situated on buildings from those on trees can be a difficult task when trees are close to buildings and have branches cover part of the buildings. Here we discuss the simpler situation from only building extraction purpose. We assume buildings and trees described the different jump edges from step 2. Then we need to indentify which jump edges represent buildings. In other words, the buildings and trees are separated by theirs outlines. Generally, this goal can be achieved by knowledge-based analysis. In term of size, shape and linearity, the outlines of buildings are quite different from ones of trees.

In this research, we use the dimensionality learning method to separate the buildings and trees. The inner complexity of a point clouds can be capture by its intrinsic dimension. Here intrinsic dimension is defined as the correlation dimension. For a finite set $S_n = \{p_1 ... p_n\}$ in a metric space, let

$$C_n(r) = \frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{j=i+1}^{n} 1: if\ distance(p_i, p_j) < r \qquad 1$$

where, r is the search radius. If the limit exists, $C(r) = lim_{n\to\infty} C_n(r)$, the correlation dimension of S is defined as:

$$D_{corr} = \lim_{r\to 0} \frac{logC(r)}{logr} \qquad 2$$

For a finite sample, the zero limits can't be achieved. By plotting logCn(r) against logr, the correlation dimension is estimated by measuring the slope of the linear part. **Figure 10** shows the examples of three random generated point clouds. The first one is the points randomly picked on a circle; it has the same neighborhood structure as a straight line, so its correlation dimension is equal to 1.0. The second is the points picked from the surface of a sphere, and its correlation dimension is equal to 2.0. The third one consists of the points randomly distributed inside the sphere, its correlation dimension is 2.62, and it indicates this point cloud is closer to 3-dimensional volume than 2-dimensional surface structure.

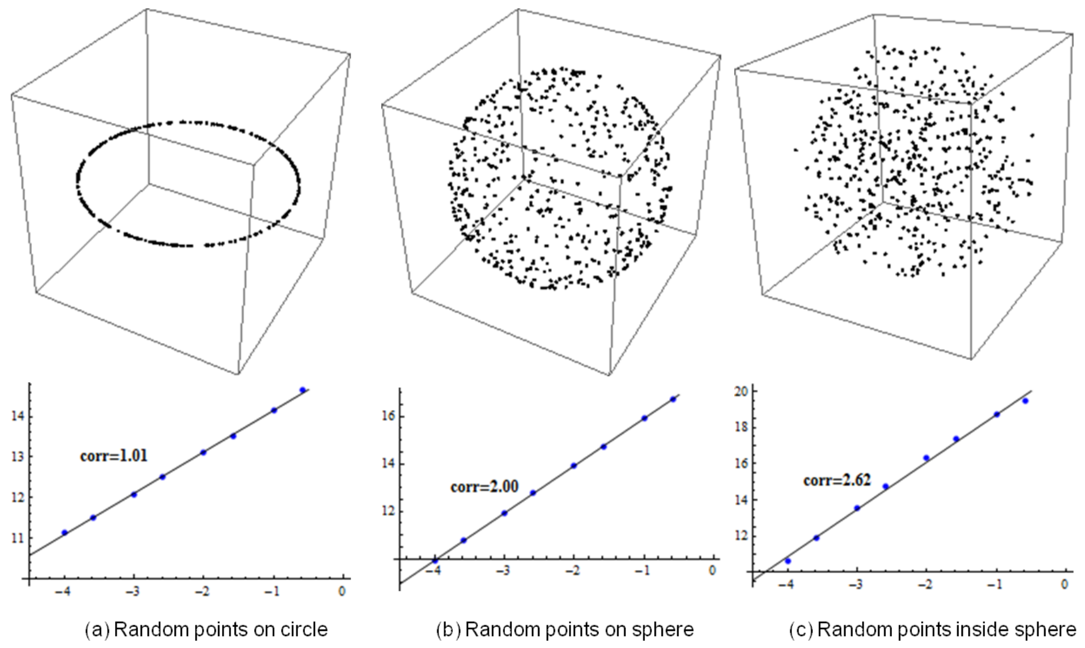


Figure 10 Examples of the correlation dimension

Intrinsic dimension is independent from the embedding space. A curve in 3-dimensional space can be very complex in appearance compared to a straight line, even though both of them have the same intrinsic dimension. **Figure 11**shows two LiDAR point clouds examples, one is a simple building, and the other is the small tree nearby. Laser beams penetrate the top surface of the small tree, the point clouds includes both points from the surface of the tree and under the canopy. The intrinsic dimension of the tree point clouds is 2.55; on the contrary, the building one is 2.08, since Laser beams only collect information on top of the building. These two point clouds are distinguishable by measuring their intrinsic dimensions.

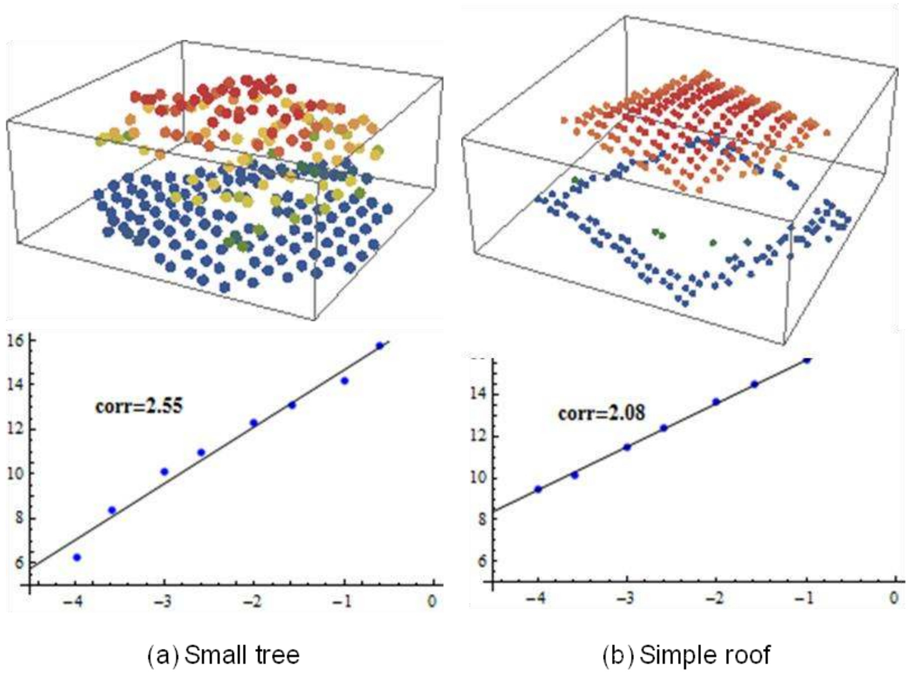
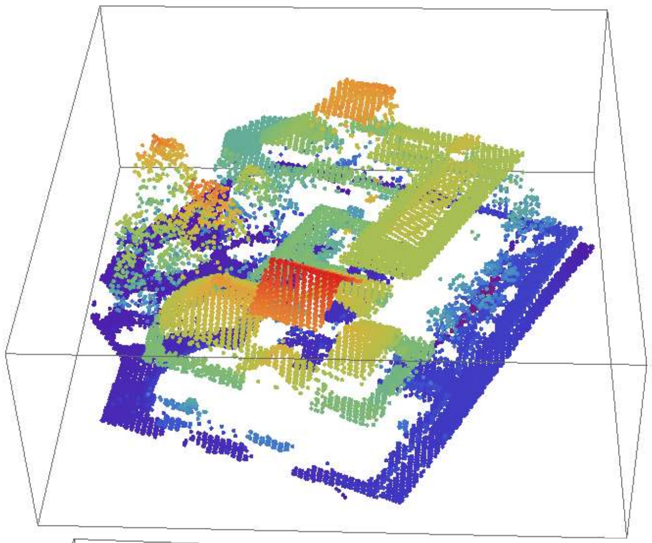
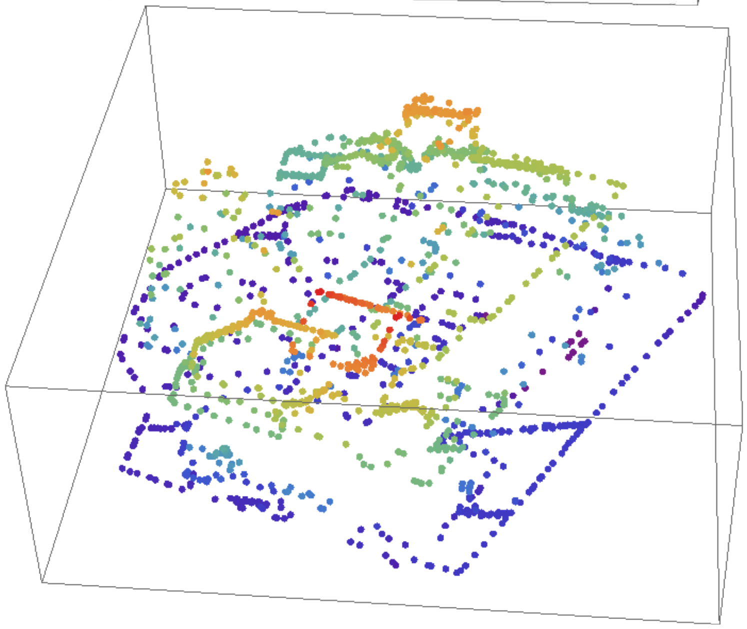


Figure 11 Correlation dimensions of two LiDAR point clouds

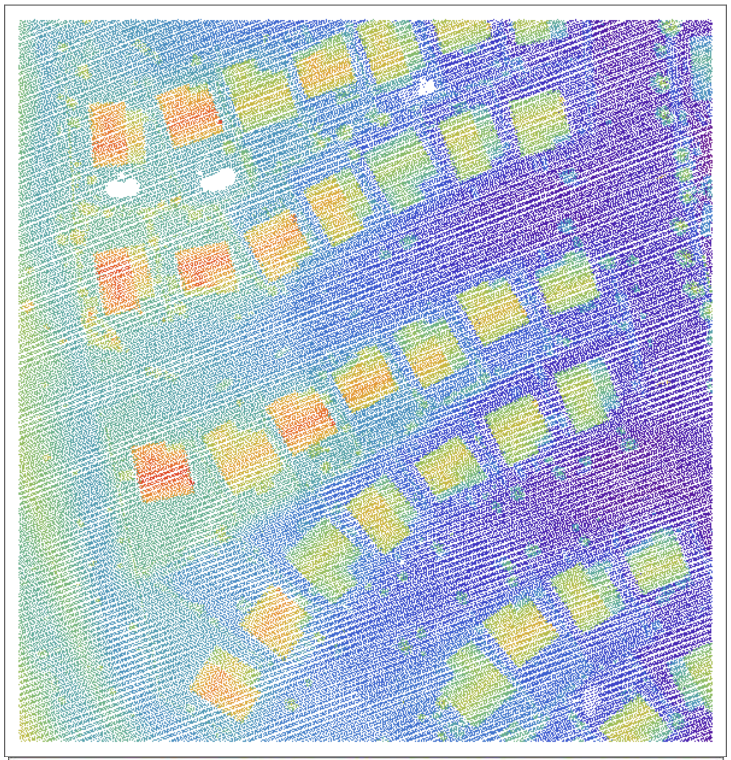
TESTING ON AIRBRONE LIDAR DATA

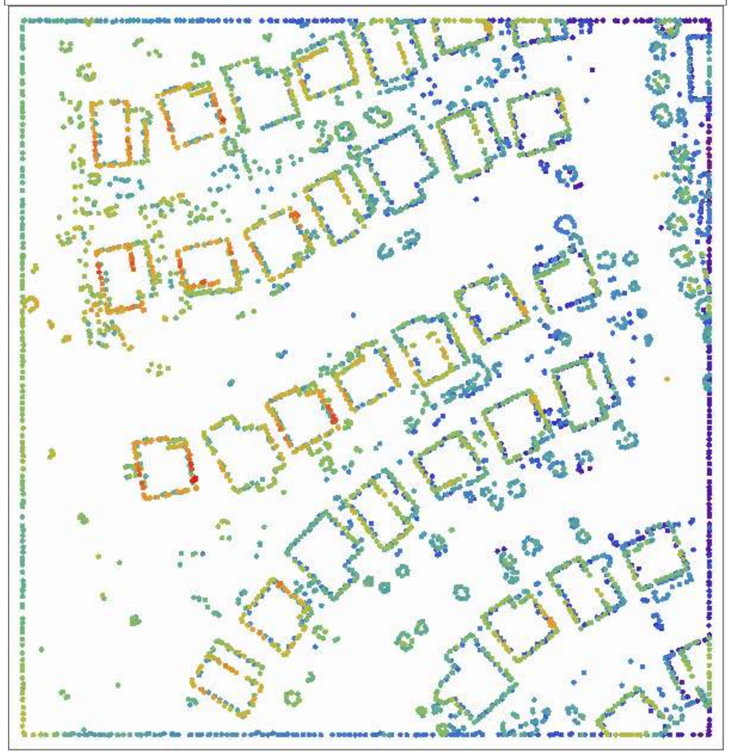
The building algorithm is tested on two sets of airborne data set. The first one demonstrates the ability to extract multi-layer roof structures in one complex building. The second one aims to test the performance to extract multi-buildings in a wide area.





(missing the final graph: jump edges)





(missing the final graph: jump edges)

REFERENCES

Ahn, S. J, 2004. *Least Squares Orthogonal Distance Fitting of Curves and Surfaces in Space*, Springer, 125 pages

Biosca, Josep Miquel and Lerma, José Luis, 2008. Unsupervised robust planar segmentation of terrestrial laser scanner point clouds based on fuzzy clustering methods, *ISPRS Journal of Photogrammetry & Remote Sensing*, 63, 84-98

Camastra, F. and Vinciarelli, A, 2002. Estimating the intrinsic dimension of data with a fractal-based method, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24, 1404-1407

Chehata, Nesrine; David, Nicolas and Bretar, Frédéric, 2008. LIDAR Data Classification using Hierarchical K-means clustering, *ISPRS Congress Beijing 2008*, 37, 325-330

Chung, F. R., 1997. Spectral Graph Theory, *CBMS Conference on Recent Advances in Spectral Graph Theory*, AMS Bookstore, 207 pages

Filin, Sagi, 2002 Surface clustering from airborne laser scanning data, *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 34, 117-124

Fuchs, F., 2001. Building reconstruction in urban environment: a graph-based approach, *Automatic Extraction of Man-made Objects from Aerial and Space Images (III)*, Baltsavias, E.P., Gruen, A. and Gool, L.V. (ed.). Taylor & Francis, 205-215

Gorte, Ben, 2002. Segmentation of TIN-structured laser altimetry points clouds, *Symposium on Geospatial Theory, Processing and Application*, 5 pages

Haxhimusa, Y. and Kropatsch, W.G, 2003. Hierarchy of partitions with dual-graph contraction, *Proceedings of German Pattern Recognition Symposium*, LNCS 2871, 338-345

Heath, M.; Sarkar, S.; Sanocki, T. and Bowyery, K, 1998. Comparison of edge detectors a methodology and initial study, *Computer Vision and Image Understanding*, 69, 38-54

Hoover, A.; Jean-Baptiste, G.; Jiang, X.; Flynn, P.J.; Bunke, H.; Goldgof, D.B.; Bowyer, K.; Eggert, D.W.; Fitzgibbon, A. and Fisher, R.B, 1996. An experimental comparison of range image segmentation algorithms, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18, 673 – 689

Lee, Impyeong and Schenk, Toni, 2002. Perceptual organization of 3D surface points, *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 34, 193-198

Jiang, X.Y. and Bunke, H, 1994. Fast segmentation of range images into planar regions by scan line grouping, *Machine Vision and Applications*, 7, 115-122

Jiang, X. and Bunke, H, 1999. Edge detection in range images based on scan line approximation, *Computer Vision and Image Understanding*, 73, 183-199

Khalifa, I.; Moussa, M. and Kamel, M, 2003. Range image segmentation using local approximation of scan lines with application to CAD model acquisition, *Machine Vision Applications*, 13, 263-274

Kim, Taejung and Muller, Jan-Peter, 1999. Development of a graph-based approach for building detection, *Image and Vision Computing*, 17, 3-14

Preparata, F. P. and Shamos. M. I, 1985. *Computational Geometry: An Introduction*, Springer, 390 pages.

Pu, Shi and Vosselman, George, 2006. Automatic extraction of building features from terrestrial laser scanning, *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36, 5 pages

Rabbani, T.; Heuvel, F. v. d. and Vosselman, G, 2006. Segmentation of point clouds using smoothness constraints, *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36, 248-253

Roggero, Marco, 2001. Airborne laser scanning: clustering in raw data, *International Archives of Photogrammetry, Remote Sensing*, 34, 227–232

Rottensteiner, F, 2003. Automatic generation of high-quality building models from lidar data, *IEEE Computer Graphics and Applications*, 23, 42-50

Sappa, A.D. and Devy, M, 2001. Fast range image segmentation by an edge detection strategy, *Proceedings of 3nd International Conference on 3-D Digital Imaging and Modeling*, 292-299

Schnabel, R.; Wahl, R. and Klein, R, 2007. Efficient RANSAC for Point-Cloud Shape Detection, *Computer Graphics Forum*, 26, 214-226

Shapiro, Linda G. and Stockman, George C, 2001. *Computer Vision*, Prentice Hall, 580 pages

Shi, J. and Malik, J, 2000. Normalized cuts and image segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22, 888-905

Sithole, George and Vosselman, George, 2003. Automatic Structure Detection in a Point Cloud of an Urban Landscape, *Proceedings of 2nd GRSS/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas*, 67-71

Vosselman, G.; Gorte, B.G.H.; Sithole, G. and Rabbani, T, 2004. Recognising structure in laser scanner point clouds, *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 46, 33-38

Wang, Lu and Chu, Henry, 2008. Graph theoretic segmentation of airborne LiDAR data, *Proceedings of SPIE*, 6979, 10 pages