IMPLEMENTATION OF 3D POINT CLOUDS SEGMENTATION BASED ON PLANE GROWING METHOD

Małgorzata Jarząbek-Rychard[1]

Abstract

In the recent years we can observe growing demand for the extraction of physical objects from 3D spatial information. Segmentation is the first step of point cloud interpretation which further allows object modeling. This paper presents an implementation of an automatic tool for 3D data segmentation. The segmentation algorithm is based on the plane growing approach, which uses least squares plane estimation and point connectivity provided by the *k*-nearest neighbours search. Two refinement parameters are required, angle and distance thresholds. In order to avoid a loss of information caused by data resampling, presented software works on raw 3D data.

Keywords

Segmentation, laser scanning, 3D modeling

1 INTRODUCTION

In the recent years, we can observe growing demand for the extraction of physical objects from three-dimensional spatial information. Laser scanning is considered a leading technology for the acquisition of such data. As well, image-matching algorithms that work on images acquired by digital cameras, experience a renaissance. Both techniques allow extracting a dense data set that contains sub-randomly distributed points, referred to as point cloud. Dense point clouds are highly desirable; however, they may present a challenge with regard to the management of huge volumes of data and processing limitations on PCs [1].

Point cloud data sets allow for the analyses of complex constructions by establishing accurate and detailed 3D models. The field of application of object modeling is very wide (e.g. industry, urban planning or water management) and still grows as new methods appear and old ones improve. Usually the modeling is used for the purpose of man-made objects extraction, such as buildings, bridges or industrial objects. Due to increasing demands for such products, numerous disciplines deal with the problem of retrieving explicit spatial information from the abundant point cloud.

The capability of carrying out useful analyses is simplified for data with some organizational structure. It becomes considerably limited for unstructured 3D points. Therefore, before final products can be derived, raw point clouds require a great deal of processing. Acquired data feature the lack of structure and implicitness of included information. Therefore, the problem of automatic extraction of the shape of the recorded objects becomes highly complex. Accounting to both effectiveness of extracting information and the constantly increasing number of data, there is the necessity to introduce some level of organization into a data set [2]. Such organization involves data segmentation that provides planar faces (for instance in case of building modeling) or specific geometric shapes by aggregating points with similar attributes. This process enables afterwards to reconstruct geometric 3D models. For this reason, the segmentation process emerges as one of the most crucial part of automatic point clouds processing. The importance of this task is clearly visible in the number of segmentation algorithms developed in the last few years.

Since many segmentation algorithms have their origination in range image analyses, they require 2.5D input data. Because raw data acquired by laser scanning or multiple image matching feature 3D character, they have to be transformed to a raster. Such a transformation usually causes loss of information. Therefore, the current research leans to improve algorithms that are working on raw 3D data. One of these approaches – plane growing based segmentation – is used for the software described in this paper.

2 SEGMENTATION

Segmentation is the first step of point cloud interpretation. It allows partitioning of a data in the object space in order to create meaningful, coherent and connected subsets, referred to as segments. Each segment is supposed to include points or pixels with similar attributes and represents a surface, an object or a part thereof. The outcome of the segmentation is the set of extracted segments.

[1] Małgorzata Jarząbek-Rychard, M.Sc., Wroclaw University of Environmental and Life Sciences, The Faculty of Environmental Engineering and Geodesy, Institute of Geodesy and Geoinformatics, Grunwaldzka 53, 50-357 Wroclaw, malgorzata.jarzabek-rychard@up.wroc.pl

Denoting whole data set as R and each segment as R_i the segmentation process may be formally described as follows [3] :

1) $\cup_{i=1}^{n} R_i = R$,
2) R_i is a connected segment, $i = 1, 2, ..., n$,
3) $R_i \cap R_j = \{\}$ for all i and j, $i \neq j$,
4) $P(R_i) =$ TRUE for all $i = 1, 2, ..., n$,
5) $P(R_i \cup R_j) =$ FALSE for $i \neq j$, where P denotes probability.

The whole data set is decomposed into segments (1) that are disjoined (3). Each of them is a connected component in the object space (2), which means that grouping has to take into consideration a neighbourhood between the points. The predicate (4) defines the similarity among the points providing that similar points will lie on the same object plane. Requirement (5) ensures that the points belonging to different segments lie on separate faces.

The features that are considered while partitioning a point cloud may be either geometrical or spectral. However, the intensity data that provided by laser scanning has noisy character. Due to this fact, points are usually grouped with respect to spatial properties only.

2.1 Plane growing method

A segmentation method used for the presented implementation is based on extraction of planar faces from unstructured data sets, using the scheme of the plane growing approach. The algorithm is driven by the best-fitting plane estimation. The aim of segmentation process, i.e. smooth surfaces extraction is achieved by grouping neighbouring points that share some properties, such as the direction of a locally estimated surface normal. In this work, the processed data is a raw unstructured 3D point cloud. The assumption about this kind of input makes presented method more general.

The presented algorithm consists of the following core steps: structuring data using nearest neighbours search, computing the best-fitting plane, the plane growing process. Ultimately, segment merging is performed in order to avoid over-segmentation.

2.1.1 Nearest neighbours search with usage of a *k*-D trees structure

The nearest neighbour (NN) search problem requires preprocessing a data set into a structure that provides a quick finding of the closest point to a query point in the metric space [4]. This idea might be easily extended to searching for *k*-nearest neighbours. The principle of this approach is based on geometrical relations between points.

In order to optimize search time performance many optimization strategies have been developed in the recent years. One of them is space partitioning [5]. *k*-D tree is one of space partitioning data structure for storing a finite set of points from a *k*-dimensional space. It is a special kind of a binary tree - which is the result of recursive subdivision of the three dimensional space [4]. Space partition and its associated node structure are outlined in Fig. 1.

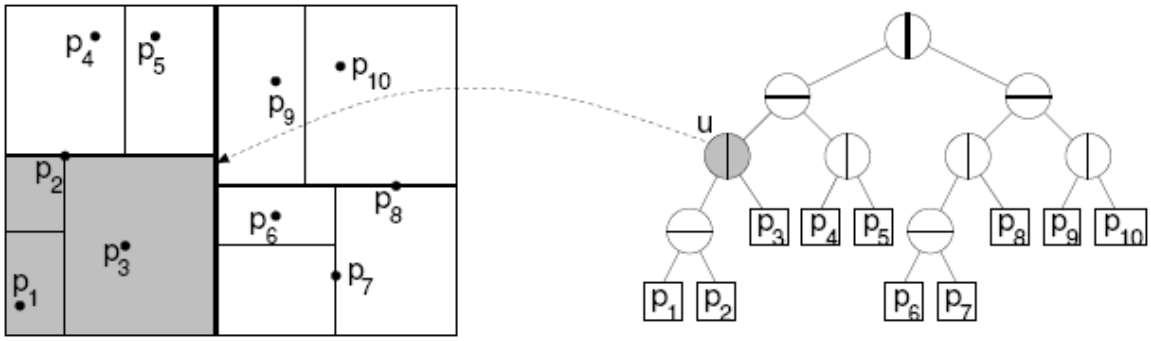


Fig. 1 *Two dimensional k-D tree - space partition and associated node structure [6].*

The idea of the nearest neighbours search algorithm operating on *k-D* trees structures consists on visiting cells successively, according to the increasing distance between them and the considered point. The first met point is considered as the nearest neighbour. The distance between this point and the query one determines the radius of the hyper-sphere centered in the query point. Consequently, any other point replacing initially approximated nearest neighbour, needs to be closer, i.e. within the volume enclosed be the hypesphere. Accounting for time performance, the sequence of point examination is based on parent-child dependency. The k-nearest neighbour search is the preprocessing step for the normal vectors estimation.

2.1.2 Least squares plane fitting

To fully define a position of the plane in the 3D space, the coordinates of the points and the plane normal have to be known. Therefore, the derivation of normal vectors is of crucial importance for the further steps of data segmentation. Plane normals should be computed and associated with each point of the data set. The calculation is performed by extraction of the best-fitting plane through the considered point and its nearest neighbours. This may be done by a least-squares fit. In terms of fitting the plane in 3D space, multiple linear regressions is applied, i.e. estimating the coefficients that provide the minimum sum of squares residuals [7].

For a given set of point coordinates, the set of parameters that minimizes the square sum of the orthogonal distances of the points from the plane has to be found [8]. A plane in 3D space may be formulated subject to

$$Ax + By + Cz + D = 0\,, \tag{1}$$

With the parameters *A, B, C* defining the plane normal vector:

$$\boldsymbol{n} = (A, B, C)\,. \tag{2}$$

According to the condition of minimizing the sum of squared distances between the i-th point $P_i(x_i, y_i, z_i)$ and the plane

$$\sum_{i=1}^{N} d_i^{\,2} \overset{!}{=} \min\,, \tag{3}$$

$$\sum_{i=1}^{N} d_i^{\,2} = \sum_{i=1}^{N} (Ax_i + By_i + Cz_i + D)^2\,, \tag{4}$$

a nonlinear least squares problem has to be solved. The problem can be reduced to an eigenvalue problem

$$(\boldsymbol{M} - \lambda\, \boldsymbol{I})\boldsymbol{x} = 0, \tag{5}$$

where $\boldsymbol{M}$ represents the coefficients matrix yielded from minimization and $\boldsymbol{I}$ is an identity matrix. As the eigenvalue is the sum of squared residuals, the best-fitting plane is associated with the smallest obtained eigenvalue. The gained set of parameters *A, B, C* is defined by the corresponding eigenvector $\boldsymbol{x}$.

2.1.3 Plane growing

Plane growing incorporates the previously computed local surface properties in order to merge together points that are spatially close and have similar geometric features. The first step of this task is to find a seed point of the surface that has to be grown. For each point of the data set, the residuals of the fitting plane determined by this point and its nearest neighbors are computed. The point with the lowest square sum of residuals is considered a seed point and its parameters are assumed to determine a seed surface. The second step provides segments based on the former calculation of normal vectors and distances from the best-fitting plane.

The algorithm implemented in this work is affected by two parameters: α and d that respectively stem from following geometrical restrictions:

- smoothness criterion, realized by setting a maximum value for the angle between the normal vectors of the seed plane and the candidate point (α)
- proximity criterion, enforced by setting a threshold on the distance between the seed plane and the candidate point (d)

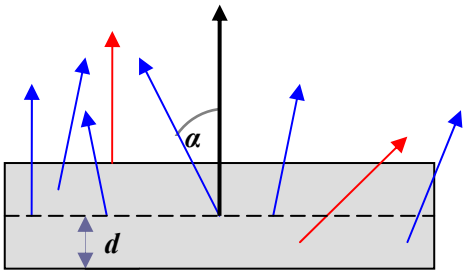


Fig. 2 *Segmentation parameters α and d; black vector: normal to the seed plane, blue vectors: accepted, red vectors: rejected*

These two requirements ensure the smoothness of the plane [9]. As a result, points are grouped into unbroken segments so that the normal vectors of points belonging to the same segment do not vary too much from each other. Figure 2 presents the idea of the algorithm.

The surface growing process starts with the seed point and continuously examines candidate points whether they fulfill or not the criterions mentioned above. This is an iterative procedure, which requires recomputing the surface equation after every loop of checking the point set and adding points. This updating enables to extract more precise plane parameters, as more points are used for their calculation. The procedure is repeated and consequently the surface is growing until no candidate meeting the criterions is found. Thus, no more point can be accepted and assigned to the segment. In this case, a new seed point has to be found and the surface growing process is carried out again.

2.1.4 Segments merging

This last stage enables to avoid over-segmentation in the final result. Starting with the algorithm execution first, the near neighbours search is performed, which returns k neighbours for each query point. In this paper k value is set to ten. Afterwards, the best-fitting plane for these points is computed and the plane normal is derived. Unfortunately, all of the neighbours do not necessarily belong to the same plane. In such a situation, the calculated normal vector deviates from the real one. The deviation increases when the more points do not lie on the plane. Also in case of surface roughness, the surface normals do not point in exactly the same direction. These two facts affect the plane growing resulting in grouping points in some numbers of very close planes instead of one. After performing each loop of examining candidate points, their plane parameters (thus the plane normal) are updated. Therefore, split planes that in reality make up one segment should have almost the same normal vectors (possible variations are depended on a pre-defined threshold). For this reason, all segments are checked again in order to merge similar segments.

The idea of segments merging is very similar to the idea of grouping raw points during the stage of plane growing. The difference is only the input - instead of 3D singular points, the already created clusters are examined. This is again done by checking the similarity of normal vectors and the orthogonal distance between two planes. Merging similar neighbouring segments yields denser and larger planes.

3 IMPLEMENTATION

3.1 Implementation environment

The basic part of the program is coded with C++. In order to extend the computational functionalities of this language, additional external libraries were included. The extension starts with incorporating two classes belonging to LASTools. They enable to read and write the processed data that are stored in the LAS (Log ASCII Standard) format. The GNU Scientific Library and the Approximate Nearest Neighbour Library simplify the implementation of segmentation algorithm. For the purpose of visualization the program uses the Open Graphics Library. To enable external interaction a Graphic User Interface was developed, which was done with the usage of the Qt platform.

3.2 Algorithm

The segmentation algorithm implemented in this work, consists of the four main steps:

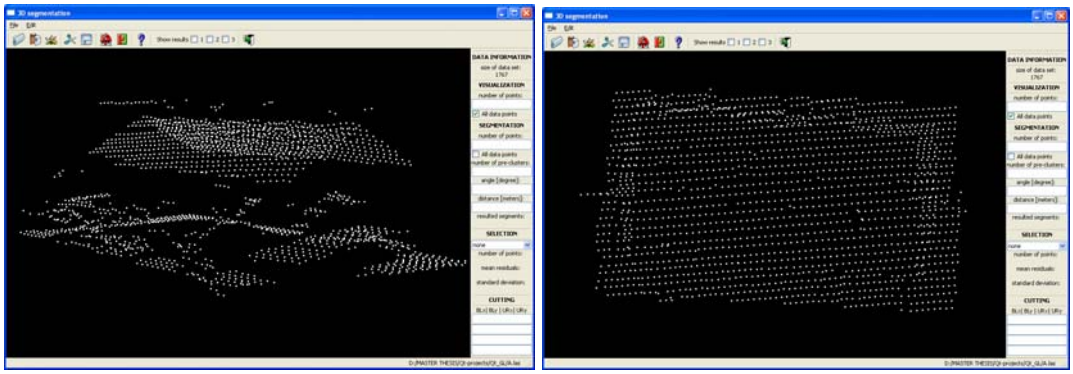
1) Data preparation :
   a) Finding for each point the n-nearest neighbours in the 3D space using k-D trees, $n = 10$.
   b) Computing normal vectors - this step is associated with deriving for each point the equation of the best-fitting plane, determined through that point and its nearest neighbours.
   c) Computing mean residual and standard deviation of the distances from points to their best-fitting plane. The respective values are associated with each point.
2) Setting the criterions the surface growing procedure is depended on and specifying thresholds associated with them.
3) Finding segments :
   a) Picking the seed point by analyzing the best fitting-plane's residuals.
   b) Performing an iterative plane growing process.
   c) When the extending process is finished, detection of a new seed point and repeating the step (a).
4) Merging homogenous segments divided into different sets (performed in order to avoid over-segmentation). The principle of this step is the same like for the third one. Instead of singular points, segments are used as an input.
   a) Picking the seed segment by analyzing the best fitting-plane's residuals.
   b) Performing an iterative plane growing process.
   c) When the extending process is finished, detection of a new seed point and repeating the step (a).

4 APPLICATIONS AND RESULTS

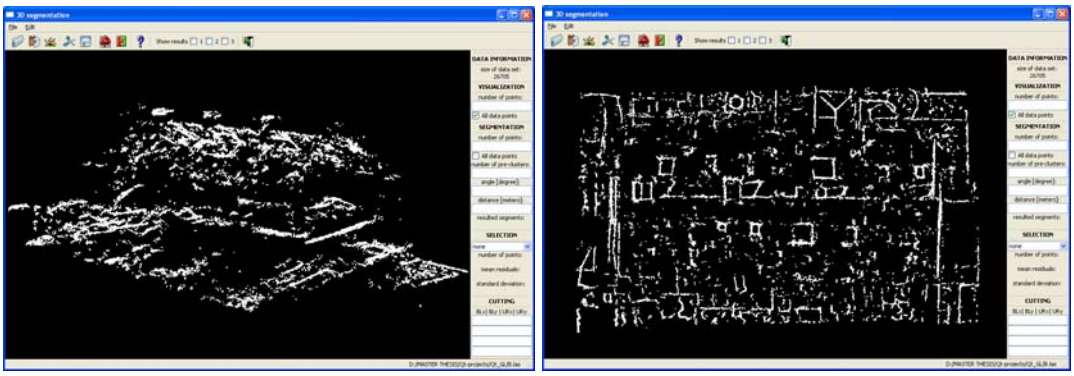
Tests are performed in order to evaluate the applicability of the implemented surface growing approach. Therefore, the algorithm is examined against point clouds derived from three sources. For the purpose of results comparison, the segmentation of each considered area is performed with the same parameter settings. Afterwards the segments that can be clearly distinguished in each outcome are chosen and visually compared. To facilitate readability, small clusters are discarded and the number of presented segments is limited to 30. Additionally, statistics computed for chosen objects enable to assess the segmentation performance. The provided values give information about the number of points within the chosen segment, mean residual from the best-fitting plane and standard deviation.

4.1 Study area and data

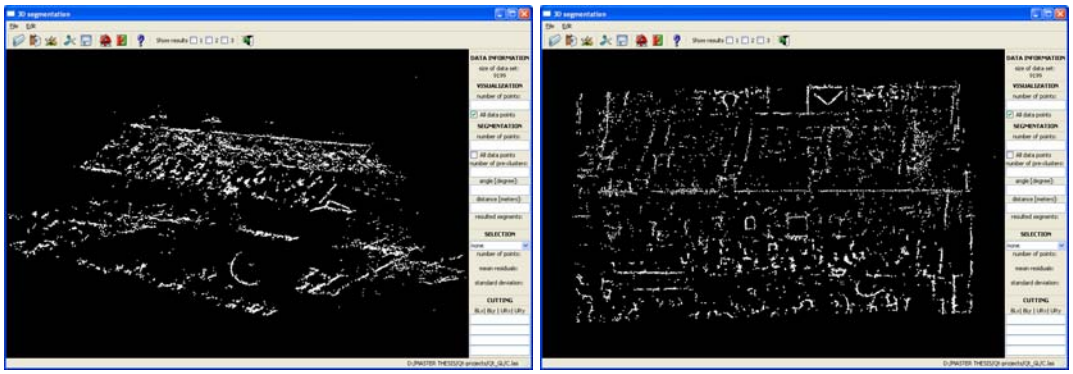
The used data consist of three point clouds. Each of them refers to the same object - relatively simple house with a gable roof. The point clouds are acquired from different techniques: airborne laser scanning and multiple image matching. LiDAR data were collected by an ALS50 system during ten flights with 45º FOV and the density of 5 points/m^2. The first matched point cloud is generated by matching images acquired by a UCX Camera. The second one is derived by processing photos delivered by DMC. Image blocks of both cameras were acquired with 8 cm GSD and 60% overlap along and across the strips. The visualization of data is presented in Fig.3.



(a) LiDAR



(b) UCX with 8 cm GSD

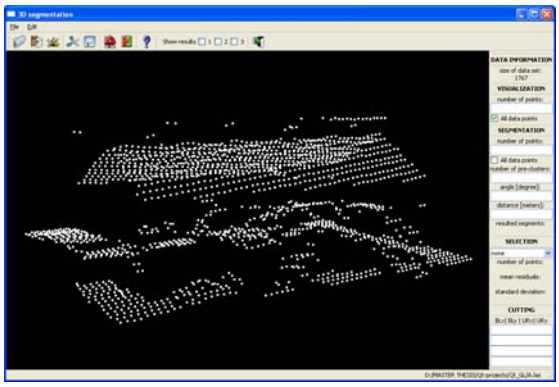


(c) DMC with 8 cm GSD

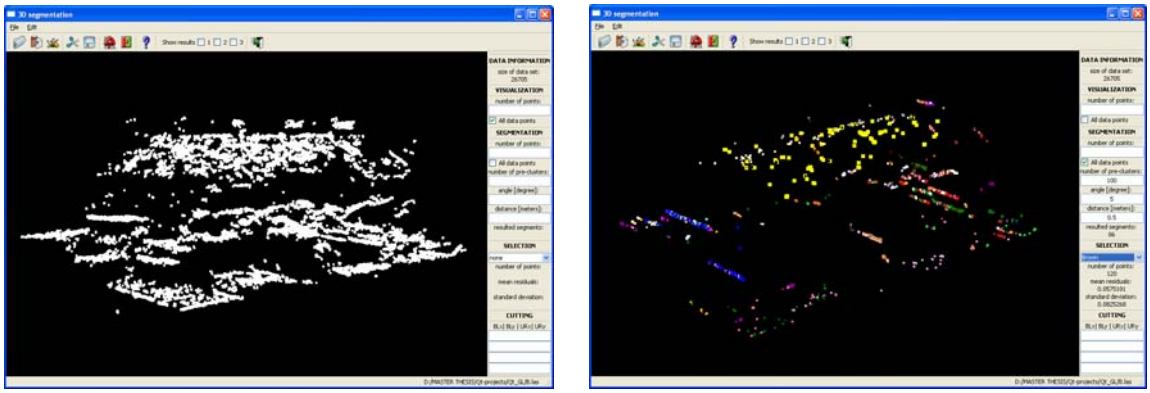
Fig. 3 *Considered building. Data provided by different techniques*

4.2 Investigation

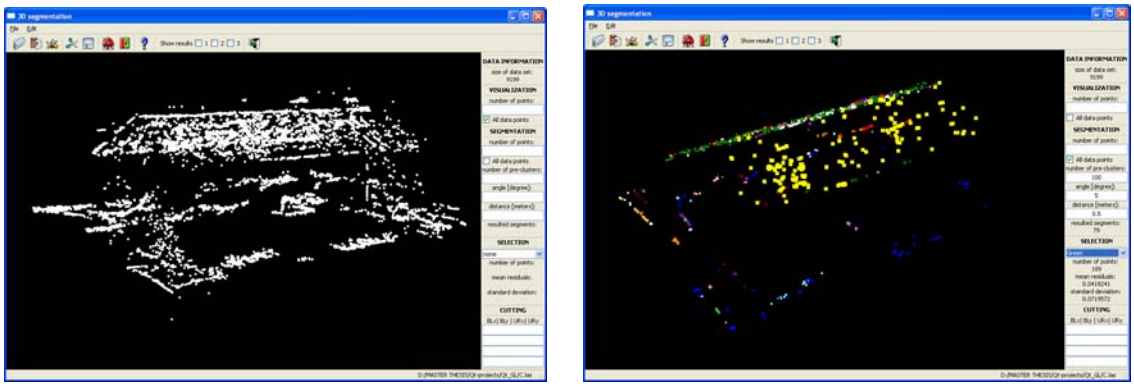
For each segmentation process, the angle threshold α was set to 5º and the maximum distance equals 0.5 m. The results of building segmentation are highlighted in Fig. 4. Since the roof segment is clearly distinguished in each data set, it was chosen to conduct the comparison of segmentation statistics. Table 1 presents the comparison of results obtained from three different sources. From the figure and the table it is seen that the most plausible results are obtained from processing airborne laser scanning data (Fig. 4(a)). Both the grouping of smooth planes and the avoiding of over- and under-segmentation have been successfully achieved in this case. Figure 4(b) outlines the outcome of processing the data acquired by the UCX Camera. Besides a roof surface, the segmentation results in extraction of short edges instead of planes. The raw points provided by image matching are not equal distributed and concentrated at discontinuities. Moreover, parts of the depicted objects are not smoothly connected. Therefore, due to the character of the data, application of the plane growing algorithm gives a poor performance. Another point cloud derived from image matching arises from photos acquired by DMC. Outcomes of its segmentation are presented in 4(c). The results are better than in the UCX case, as the point cover is denser and the point distribution is more homogeneous. However, still the quality of segmentation is not satisfying.



(a) LiDAR



(b) UCX with 8 cm GSD



(c) DMC with 8 cm GSD

Fig. 4 *Original point clouds derived from different sources (left images) and their segmentation with highlighted roof points (right images)*

Tab. 1 *Comparison of roof segments*

Data source	Number of points	Mean residual [m]	Standard deviation [m]
LiDAR	373	0.017	0.021
UCX 8	120	0.057	0.082
DMC 8	189	0.042	0.072

5 CONCLUSION

Difficulties within the implementation arose for the *k*-nearest neighbours search and normal vector computation. The latter is provided for each point, based on its *k*-nearest neighbours and their best-fitting plane. However, the selected points do not necessary lie in the same plane. In this case, the least squares plane estimation is biased, resulting in false normal vectors. Therefore, the first grouping points into a segments (called 'pre-clusters') is not sufficient. Next, it is required to update best-fitting plane's equation for each pre-cluster. After updating, plane growing has to be carried out again using pre-clusters instead of singular points.

Originally, the implemented algorithm was characterized by two segmentation parameters, angle threshold and distance threshold. Due to the problems arising for the computation of accurate normal vectors, the run time of algorithm increased considerably. To meet a tradeoff between the number of segments and the time performance, an additional refinement parameter was introduced, i.e. number of created pre-clusters. This threshold enables to skip the computation of small clusters during the segmentation process. In order to facilitate distinguishing the large number of objects in visualization process, small segments are not displayed. Focusing on large segments only, significantly improves the algorithm performance.

It can be inferred from the conducted tests, that the performance of the implemented algorithm is strongly depended on the character of the input data. Point clouds derived by multiple image matching are denser at the edges than at the continuous surfaces. Therefore, the segmentation of matched point clouds often returns clusters that depict several parts of object instead of one whole object. However, among matched point clouds there are some that suit much better than others do. In both tests, segmentation of data stemming from DMC systems give results most comparable with LiDAR counterpart. The segmentation approach is tailored for point sets that feature some level of order, like for example even distribution or regular spacing. As the points provided by LiDAR are collected strip-wise, this kind of input gives very plausible results.

Literature

[1] BIOSCA J.M., LERMA J.L. *Unsupervised robust planar segmentation of terrestrial laser scanner point cloud based on fuzzy clustering methods.* ISPRS Journal of Photogrammetry and Remote Sensing, 63, 2008, pp. 84-98.

[2] FILIN S., PFEIFER N. *Segmentation of airborne laser scanning data using a slope adaptive neighborhood.* ISPRS Journal of Photogrammetry and Remote Sensing, 60, 2006, pp. 71-80.

[3] HOOVER A., JEAN-BAPTISTE G., JIANG X., FLYNN P.J., BUNK, H., GOLDGOF D.B., BOWYER K., EGGERT D.W., FITZGIBBON A., FISHER R.B. *An Experimental Comparison of Range Image Segmentation Algorithms.* IEEE Trans. Pattern Analysis and Machine Intelligence, 18, No. 7, 1996, pp. 673-689.

[4] GOODMAN J.E., O'OURKE J. *Handbook of discrete and computational geometry*. CRC Press, Inc., Boca Raton, New York, 1997.

[5] MOORE A.W. *An introductory tutorial on k-D trees.* Extract from PhD Thesis: Efficient Memory-based Learning for Robot Control, University of Cambridge, UK, 1991.

[6] MOUNT D.M. (2006) *ANN Programming Manual* [online] http://www.cs.umd.edu/~mount/ANN/Files/1.1.1/ANNmanual_1.1.1.pdf

[7] CHAPRA S.C., CANALE R.P. *Numerical methods for engineers: with software and programming applications*. The McGraw-Hill Companies, Inc., New York, 2002.

[8] WANG M., TSENG Y-H. *LiDAR data segmentation and classification based on OCTREE structure*. Geo-Imagery Bridging Continents 20th ISPRS Congress, Istanbul, Turkey, 2004.

[9] TOVARI D. *Segmentation based classification of airborne laser scanner data.* PhD Thesis, University of Karlsruhe, Germany, 2006.

Reviewer

Andrzej Borkowski, prof. dr hab. inż., Wroclaw University of Environmental and Life Sciences, The Faculty of Environmental Engineering and Geodesy, Institute of Geodesy and Geoinformatics, Grunwaldzka 53, 50-357 Wroclaw, Poland, +48713205609, andrzej.borkowski@ up.wroc.pl