An Approach for Segmentation of Airborne Laser Point Clouds Utilizing Scan-Line Characteristics

Soo-Hee Han, Jeong-Ho Lee, and Ki-Yun Yu

In this study, we suggest a new segmentation algorithm for processing airborne laser point cloud data which is more memory efficient and faster than previous approaches. The main principle is the reading of data points along a scan line and their direct classification into homogeneous groups as a single process. The results of our experiments demonstrate that the algorithm runs faster and is more memory efficient than previous approaches. Moreover, the segmentation accuracy is generally acceptable.

Keywords: LIDAR, segmentation, classification, building extraction, point cloud.

I. Introduction

During the last decade, accuracy has been a major issue in segmenting airborne laser point clouds. Traditionally, many experiments have been based on various filtering algorithms applied to range images created by resampling point attributes [1]-[3]. Usually, these approaches intrinsically degrade positional accuracy and induce loss of the edge discontinuity characteristic. Therefore, several algorithms were applied to the raw data points without implementing the resampling [4]-[6]. As one notable example, Sampath [5] divided ground from non-ground objects with one-dimensional filtering between two consecutive points along a scan line. Although effective, this approach can be complicated when separating individual non-ground objects for which seed point selection and region growing are adopted.

Research is currently moving its focus from effectiveness to efficiency in processing as laser instruments acquire a faster pulsation frequency and greater precision. Processing speed notably attracts considerable attention because many practical applications must handle huge amounts of data. Furthermore, as the demand to construct and update building databases from remotely sensed data in urban areas increases, how to extract each building separately in a mass-productive way has become an important issue. Until now, however, very few researchers have shown interest in this issue. Sithole [4] tested the speed of proposed segmentation algorithms, in which a point cloud was partitioned to yield a series of profiles lying at different orientations. Each profile was then segmented to produce line segments and the overlaying line segments were connected to create surfaces. Iterative processes of classification and segmentation were then conducted to classify bare earth and detailed objects. This approach is targeted to extract special

Manuscript received Nov. 30, 2006; revised June 26, 2007.

This research was supported by the SNU SIR BK21 Research Program funded by the Ministry of Education & Human Resources Development.

Soo-Hee Han (phone: + 82 2 880 7371, email: scivile2@snu.ac.kr), Jeong-Ho Lee (email: ilome79@snu.ac.kr), and Ki-Yun Yu (email: kiyun@snu.ac.kr) are with the School of Civil, Urban and Geosystem Engineering, Seoul National University, Seoul, Rep. of Korea.

objects and enhance the processing speed, but it involves a memory-intensive operation including the production of a series of profiles.

Building upon this understanding, we propose a new segmentation algorithm which is faster and does not require much memory. Data points are read consecutively along scan lines and are directly classified into homogeneous groups in one single process. Eventually, segmented groups potentially mean individual buildings, their parts, vegetation, bare earth, and other divisions. Although classification, as a next step, is needed for clarification, this paper focuses on efficient segmentation. We are planning a future presentation on the classification routine appropriate for the proposed segmentation algorithm.

To test the validity of the proposed algorithm, a typical single strip of a point cloud and the ISPRS filter test set [7] were studied and processed. The processing time and memory handling were then examined, and an accuracy assessment was also carried out.

II. Proposed Segmentation Algorithm

The basic algorithm of the proposed approach is similar to the region-growing and unsupervised-classification methods. A first input point is read from a data file; thus, constituting a new group. Starting with the next input point, each point is compared with the previous ones. If the new input point is adjacent to a previous point having a similar height, it is classified into the group to which the previous point belongs; otherwise, it becomes a new seed point generating a new group.

To make this algorithm effective, points need to be read along a scan line to increase adjacency. In this approach, therefore, all points should be arranged along their scan lines during preprocessing if they are not already so arranged. In the sense that linearly arranged pieces of data are labeled according to their attributes, our approach partly resembles connected-component labeling in a raster domain [8],[9], but the scan line does not offer such regular geometry as raster data.

Concurrently, with regard to memory handling and processing efficiency, we also devised three strategies that work regardless of data size.

1. Basic Algorithms

The basic segmentation routine is carried out as follows.

Step 1. Set n=1
Step 2. Empty out L_{merge} and read p_{new}
Step 3. For i=1 to n
  For j=1 to m_i {
    Calculate $d_{new,ij}$ and $h_{new,ij}$ between p_{new} and p_{ij}
    If $d_{new,ij} \le t_{dist}$ and $h_{new,ij} \le t_{height}$ {
      If p_{new} is not classified
        Classify p_{new} into G_i
      Put G_i into L_{merge}
    }
  }
Step 4. If num(L_{merge})=0{
  n=n+1
  Create G_n and classify p_{new} into G_n
}
Else if num(L_{merge})>1
  Merge groups in L_{merge}
Step 5. If there are more p_{new} to classify, go to step 2
Else exit

Here, n is the number of the point groups, m_i is the number of member points in point group G_i, L_{merge} is the list of groups to be merged, p_{new} is a new point to be classified, p_{ij} is the j-th member point of point group G_i, $d_{new,\,ij}$ and $h_{new,\,ij}$ denote the Euclidian distance and height differences between p_{new} and p_{ij}, and t_{dist} and t_{height} denote the thresholds of distance and height difference.

From the above routine, all member points of every group are examined against a new point as in step 3. If the distance and height differences between the two points are within given thresholds, t_{dist} and t_{height}, the new point is classified into the group as in step 3; otherwise, it is classified into a new group as in step 4. For example, in Fig. 1, member points of group 2 have the minimum distance d_1 from p_{new}, which is larger than t_{dist}. Some of the member points of groups 1 and 4 are adjacent to p_{new} within t_{dist}, but they are lower or higher than p_{new} by h_3 and h_4, which are larger than t_{height}. Thus p_{new} is classified into group 3, which has a member point satisfying the above two conditions. In this way, five new input points after p_{new} are

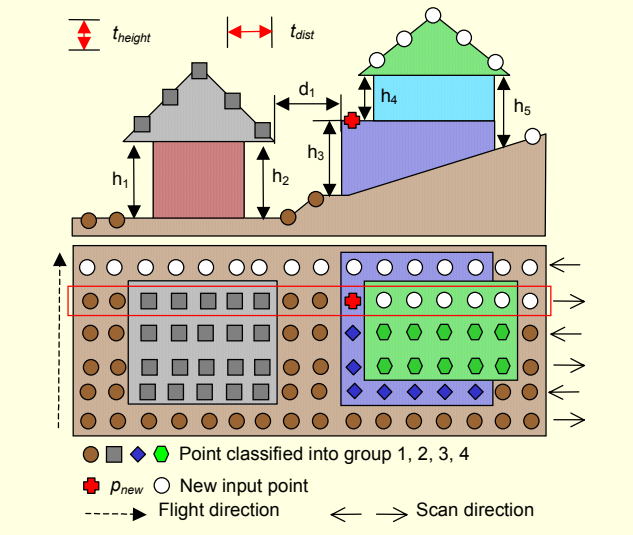


Fig. 1. Classification of a new point.

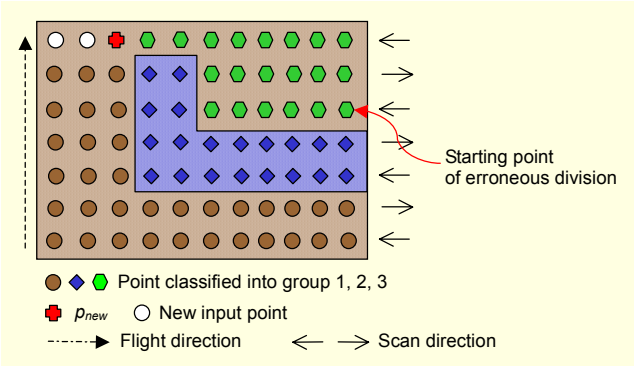


Fig. 2. Detection of erroneously divided groups.

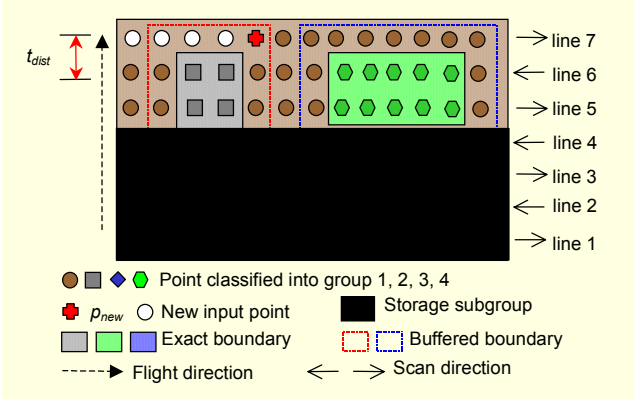


Fig. 3. Control of query.

classified into group 4 and the succeeding ones are also classified into adequate groups. Here, the upper section in Fig. 1 describes the cross-section of the red-bounded part in the vertical view of the scene.

Exceptionally, a homogeneous group may be erroneously divided into several parts. This may result from the shape of certain surrounding or intervening objects and a particular scan direction. For example, as in Fig. 2, when a building (group 2) is surrounded by earth (group 1), a new group (group 3) is created when classifying the starting point of the erroneous division, which actually belongs to earth (group 1).

In this case, p_{new} can be a key used to detect the erroneous division (Fig. 2). When p_{new} can be classified into both group 1 and group 3, the two groups can be said to be separated from a group. In this case, L_{merge} is employed so that groups into which p_{new} can be classified are put into L_{merge} in step 3. If num(L_{merge})>1, groups in L_{merge} are recognized to be divided from the same object and then merged together in step 4.

The values of t_{dist} and t_{height} cannot be determined systematically because they depend on local conditions of the dataset, such as point density, minimum distance and height differences between objects, and their complexity. Therefore, the two values should be estimated following a preliminary inspection of the dataset and should be set at values intended to reduce overdivision or overmergence of groups. Thus, t_{dist} can be generally established as a larger-than-average distance between two adjacent points and a smaller-than-minimum gap between two separable objects, and t_{height} can be set at less than the minimum height difference between two objects.

As an extension of the basic algorithms, it is possible to establish topology among segmented groups. For example, when a new point is classified to a group but is not further than t_{dist} from some member points of other groups, the groups are recognized as neighbors and then further relationships, such as the relative height of their positions can be analyzed. We are planning to utilize this concept in the future to enhance the algorithm and to develop the classification routine.

2. Three Strategies for Memory Handling and Processing Efficiency

We adopted a linked-list structure for storing groups and their member point information aiming at efficient memory management. However, as the volume of data increases, a time delay in querying groups and member points grows significantly. To solve this problem, three strategies utilizing scan-line characteristics are proposed below. The second strategy is also helpful in controlling main memory occupancy.

A. Strategy to Control Query Range for Member Points

In Fig. 3, scan line 7 is being classified, and earlier scan lines 1 to 6 are under query for their classification. However, there is no need to query scan lines from 1 to 4, and little need for scan line 5, because they are far from the input points by more than t_{dist}. It is not generally necessary to consider old scan lines further than t_{dist} from the current input. Only recently classified n_{SL} scan lines require query.

If the average gap between two scan lines is given as d_{SL}, the value of n_{SL} can be determined as $n_{SL} \times d_{SL} >> t_{dist}$. However, considering an irregular shape and arrangement of scan lines, n_{SL} should be set great enough, and 3 to 5 is judged to be suitable from experimental experience. To test the idea, we divided each group into query and storage subgroups. A query subgroup retains member points on n_{SL} newer scan lines and is queried for classification. A storage subgroup stores member points lying on older scan lines which are distant by more than t_{dist} from the current input scan line and are not further queried. In Fig. 3, supposing n_{SL} is set to 3, points on scan lines from 5 to 7 are queried for classification and points on scan lines from 1 to 4 are stored in each storage subgroup. If a new scan line 8 begins to be queried, points on scan line 5 will be moved to each storage subgroup. Thus, the query range for member points and their memory occupancy can be controlled throughout the process.

B. Strategy to Store Groups not Concerned with Classification Query

If a group no longer needs to be queried for classification, that is, if its query subgroup is empty, the points in its storage subgroup may be moved to a fixed disk with a group ID and the group information removed from the main memory. In this way, the memory requirement for storing processed data can be reduced, and meaningless referral time to groups no longer concerned with classification query can be eliminated. In Fig. 3, group 3 has no member points in its query subgroup; the member points are recorded on a fixed disk with ID 3 and all data for group 3 is removed from the main memory. In real conditions, if earth is connected throughout the scene, its query subgroup will be kept extant; thus, it will not be removed from the main memory. This will increase main memory occupancy steadily to store more earth points, but the algorithm can still be said to be memory-effective because it does not require a full set of input points organized in the main memory, and the majority of processed group data is removed from the main memory during the process.

C. Strategy to Control Query Range for Groups

For each group, a buffered boundary with width t_{dist} is updated whenever a point is classified into it. If a new input point does not fall within the buffered boundary, the group is skipped in the classification query because the new input point cannot be nearer than t_{dist} to any member points of the group. We adopt a simple rectangular boundary for speed in calculation. In Fig. 3, because p_{new} falls in the buffered boundary of groups 1 and 2 and not that of groups 3 and 4, the classification query is not applied to the latter groups.

III. Experiments and Result Analysis

The proposed algorithm was applied to two kinds of real data. The first experiment used data from the city of Daejeon, Korea, which has a single strip of an airborne laser point cloud retaining its scan-line properties. With the dataset, we tested the usefulness of the three strategies and the performance of building segmentation. The other experiment used the ISPRS test data provided by the ISPRS Working Group III/3 [7], which was originally designed to compare filtering algorithms for extracting DEMs from a point cloud. We adopted the data to test the efficiency of our algorithm under various conditions of data and parameters.

1. Daejeon Data Test

The subject site shows the typical aspects of modern cities with many residences and other buildings, some vegetation, and continuously connected bare earth with slight undulations. As the point cloud retains scan line characteristics and is composed of long strips that cover a large area, we selected one of the strips to produce a dataset. The specifications of the dataset and parameters are seen in Table 1.

Table 1. Specifications of Daejeon dataset.

Model	ALS ALTM 3070 system (Optech, Inc.)
No. of points	3 million
Point density	1.5 points/㎡
Formation of scan line	840 points/scan line
Parameters	t_{dist}= 2 m, t_{height}= 1.5 m, n_{SL}=3 lines

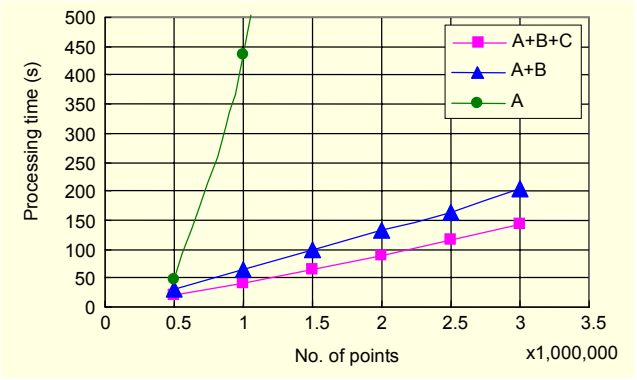


Fig. 4. Processing time comparison.

Performance assessment was conducted on different combinations of the three strategies. The required time was checked for the overall process, from reading the data file to writing the results to a fixed disk.

We observed that it took more than 3,600 seconds to process 0.5 million points without any strategy. With strategy A, it took only 46 seconds for the same points, but it took 436 seconds for 1 million points and 1,239 seconds for 1.5 million points, showing a nonlinear increment of processing time as seen in Fig. 4. Combined with strategy B, it required much less time with an increase in processing time which was linearly proportional to the size of the dataset (R^2=0.9992). Finally, when all three strategies were combined, there was about a 30% enhancement of performance over the previous combination: 144 seconds were required to process 3 million points as seen in Fig. 4.

As expected in strategy B, the main memory occupancy of the application increased during the process because the earth is connected through the dataset. It took about 38 MB at the peak to process 1 million points and about 112 MB at the peak to process 3 million points. To evaluate the validity of the memory efficiency, we made an application by simply loading the same 3 million points into the main memory. The variable

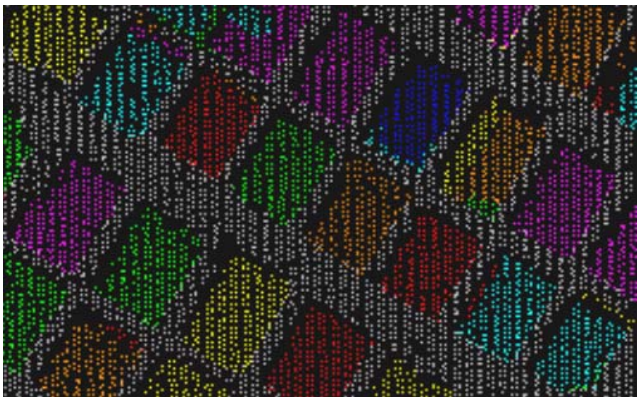


Fig. 5. Segmentation result of a building zone.

Fig. 6. Reference image of a building zone.

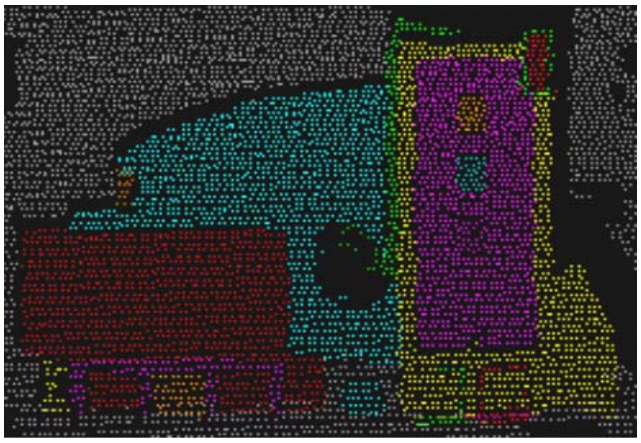


Fig. 7. Segmentation of complexly shaped building.

types for the stored points and the storage method, namely, linked-list, came exactly from the segmentation codes. Tests showed 130 MB of peak main memory occupancy and 153 MB to store the same points with labels. This result means that our algorithm requires less memory than is required to store all of the input data and is, therefore, very memory-efficient. The test system was comprised of an AMD Athlon XP 2800+ CPU with 1 GB RAM.

Fig. 8. Reference image of a complexly shaped building.

Table 2. Accuracy assessment.

Type of building	Samples	Segments	Rate
Low-rise residence	311	286	92%
Mid-high-rise building	146	146	100%

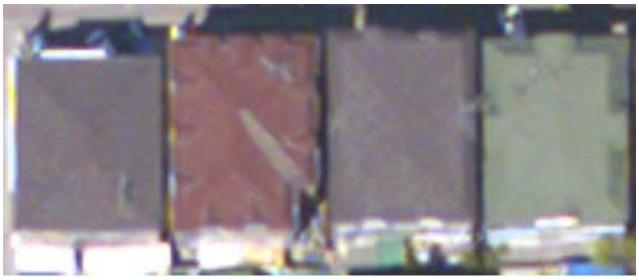


Fig. 9. Closely neighboring residences of similar heights.

Segmentation results for a building zone are illustrated in Fig. 5 with its corresponding reference image shown in Fig. 6. A visual inspection shows each building set completely apart from others and distinct from the surrounding ground.

Some complexly shaped objects can be separated into several parts if they have sections at different heights. An example of this is shown in Fig. 7 with its reference image as shown in Fig. 8. It has parts with various heights. This characteristic can be utilized for further detailed building reconstruction.

An accuracy assessment was made for building samples. According to their height, buildings are classified as low-rise and mid-high-rise. The results are summarized in Table 2. Here "segmented" means that the area of the buildings is clearly segmented in visual inspection.

From this table, it can be seen that all the mid-high-rise

buildings may be segmented well, whereas 8% of the low-rise residence buildings fail to be segmentable. Such failure is mainly due to their close mutual proximity and similar height as shown in Fig. 9.

2. ISPRS Data Test

The ISPRS data is composed of eight datasets, and they have been chosen to compare filtering algorithms because they contain various characteristics expected to cause difficulty for automatic filtering. Reference information is also provided for some subsets of each site; this information is divided into two parts: object and bare earth. We adopted the data to show the efficiency of our algorithm when tested on differently conditioned data and parameters.

The summary of the dataset is seen in Table 3. The first character of each dataset indicates where it came from: 'C' means city areas, and 'F' means forest or areas with more vegetation. We tested four parameter settings for each dataset, and the details are shown in Table 4.

To compare the performance with the research conducted in [4], we tested the algorithm in an older system, a Pentium III, 800 MHz with 256 MB RAM, equivalent to the system used in [4], which was an AMD 800 MHz with 256 MB RAM.

Table 5 shows the time spent processing each dataset under four parameter settings. The first column of each test is the real time lapse for processing given points, and the second is the expected time for processing 0.5 million points. From the results, it can be concluded that the parameters do have an effect on processing time, but not a significant one, and a dominant factor is the size of segmented point groups. More points in a group lead to more delay in finding the points lying on old scan lines that need to be moved to the storage subgroup described in strategy A. In this sense, FSites could be processed faster than CSites because FSites contain fewer buildings but more trees and consist of non-overlapping strips. This results in fewer points in each segmented group. This agrees with the inference that CSite 2 and CSite 4 took more time to process because they have relatively larger buildings than CSite 1 and CSite 3.

Additionally, when tested on the Daejeon dataset, processing took about 85 seconds for 0.5 million points. This was

Table 3. Summary of ISPRS dataset.

	No. of points	Point density	Formation of scan line
CSite1	683,204	Approx. 0.67 points/㎡	135 points/scan line
CSite2	243,400		
CSite3	188,514		
CSite4	259,030		
FSite5	314,288	Approx. 0.18 points/㎡	160 points/scan line
FSite6	275,849		
FSite7	196,632		
FSite8	172,983		

Table 4. Parameter settings.

	CSite1-CSite4			FSite5-FSite8		
	t_{dist}	t_{height}	n_{SL}	t_{dist}	t_{height}	n_{SL}
Test 1	1.8	1	3	3.8	1	3
Test 2	2.0			4.0		
Test 3	2.2			4.2		
Test 4	2.4			4.4		

Table 5. Processing time (s).

	Test 1		Test 2		Test 3		Test 4	
CSite1	77	56	81	59	81	59	79	58
CSite2	32	66	33	68	33	68	33	68
CSite3	18	48	19	50	19	50	19	50
CSite4	34	66	35	68	35	68	35	68
FSite5	21	33	21	33	21	33	20	32
FSite6	17	31	16	29	16	29	15	27
FSite7	11	28	10	25	10	25	11	28
FSite8	11	32	10	29	10	29	11	32

Table 6. Segmentation error rate (%).

	Test 1	Test 2	Test 3	Test 4
Samp11	13.99	17.24	23.21	26.66
Samp12	6.07	7.17	9.36	10.68
Samp21	6.75	7.82	9.27	10.3
Samp22	5.56	9.67	10.78	10.85
Samp23	6.32	6.47	6.88	8.77
Samp24	7.29	9.17	11.31	12.09
Samp31	6.2	8.28	9.61	10.66
Samp41	1.95	2.17	2.19	2.21
Samp42	1.55	1.57	1.83	1.85
Samp51	7.43	8.6	8.89	8.99
Samp52	3.26	3.42	3.73	3.96
Samp53	2.2	2.32	2.4	2.56
Samp54	5.9	6.61	8.24	10.53
Samp61	1.53	1.65	1.69	1.71
Samp71	5.69	5.72	6	7.36

expected because the dataset has various large buildings and continuous bare earth. There is, however, still an unexpected gap in the time used for the ISPRS dataset. This can be explained because the Daejeon dataset has more than five times the points per scan line than the ISPRS datasets. This makes the query subgroup larger, as described in strategy A; thus, generating more query operation.

The main memory occupancy for each dataset of the ISPRS did not exceed 10 MB. Compared with Daejeon dataset, this low memory occupancy is related to the fact that the ISPRS datasets have the characteristics of a narrow scanning swath, discontinuities in bare earth, and not overlapping strips.

Segmentation error was checked for subsets in each dataset with the provided reference information. The result is shown in Table 6. The first digit in the sample name is the dataset number in Table 5, and the second is the subset number in the dataset. We determined the segmentation error as the state of heterogeneity. That is, if the greater part of a group is identified to be classified into class A and the remainder into class B, the error is calculated as the ratio of the number of points classified into class B and the total number of points of the group.

As shown in Table 6, the error rate is at an acceptable level except for Samp11, which contains a mixture of vegetation and buildings, data gaps, and so on. A lower t_{dist} can drop the error ratio and a similar situation is expected to be applicable to t_{height}, but it can also bring about over-fractionation of groups. In conclusion, the accuracy assessment does not have much significance at this stage. This is because an automatic classification routine executed after segmentation may lower the final accuracy in general, but delicately designed additional routines, such as iterations of segmentation and classification, suited for specific target extraction, can bring about the opposite result.

3. Performance Comparison

We compared the performance of our approach with the result in [4]. The segmentation routine adopted in [4] can be summarized as follows. A point cloud is partitioned in several directions (more than three directions for a better result) to yield a series of profiles in a given width. The point cloud is the union of all the profiles with the same orientation, such that no two profiles with the same orientation share common points, and points in the profiles are sequentially ordered. Each profile was segmented to yield line segments, and several labeling methods, including consecutive labeling, labeling by proximity, and labeling using minimum spanning trees, were tested. Overlapping line segments were then gathered from differently oriented profiles to yield surfaces. The algorithm is thus memory-intensive for producing the series of profiles, the overlapping labels for each point, and so on.

In [4], it took about 120 seconds to process 0.5 million points with consecutive labeling, which is most comparable to our algorithm because it labels points according to the planimetric and height difference between two consecutive points in a profile. With the other labeling methods, more than 50% of this processing time was expected, whereas it took at most 70 seconds for the ISPRS datasets and 85 seconds for the Daejeon datasets using our algorithm.

Furthermore, according to Sithole [4], it was expected that about 240 seconds would be required to process 1 million points with sufficient memory available. In comparision, our algorithm required 183 seconds with a maximum main memory occupancy of about 38 MB and 538 seconds with about 112 MB for 3 million points of the Daejeon dataset. This demonstrates that the proposed algorithm provides a faster and more memory-efficient approach, despite the variance of data.

IV. Conclusion

In this study, we proposed a new segmentation algorithm for airborne laser point clouds utilizing scan-line characteristics. The algorithm proved to be faster and more memory-efficient than previous approaches. It notably requires an almost linear increase in the time taken to process enormous quantities of data as the amounts increase. With regard to accuracy, experiments showed that the algorithm allowed a good segmentation of building areas, giving a generally acceptable outcome. Based upon these results, we anticipate that the method will be used for many applications when data processing on an enormous scale is needed.

For future work, we have plans to give more functions to the segmentation algorithm in supporting topology within segmented groups, and we intend to build up a classification routine to cope with this. In addition, to reconstruct detailed three-dimensional building models, we will make use of segmented parts belonging to the same object.

References

[1] J. Kilian, N. Haala, and M. Englich, "Capture and Evaluation of Airborne Laser Scanner Data," *International Archive of Photogrammetry and Remote Sensing*, XXXI, Part B3, 1996, pp. 383-388.

[2] P. Lohmann, A. Koch, and M. Schaeffer, "Approaches to the Filtering of Laser Scanner Data," *International Archives of Photogrammetry & Remote Sensing*, XXXIII, Part B3, 2000, pp. 540-547.

[3] K. Zhang, S.C. Chen, D. Whitman, M.L. Shyu, J. Yan, and C. Zhang, "A Progressive Morphological Filter for Removing Non-

ground Measurements from Airborne LIDAR Data," *IEEE Trans. Geoscience and Remote Sensing*, vol. 41, no. 4, 2003, pp. 872-882.

[4] George Sithole, *Segmentation and Classification of Airborne Laser Scanner Data*, Publications on Geodesy 59, NCG, Delft, 2005.

[5] A. Sampath and J. Shan, "Urban Modeling Based on Segmentation and Regularization of Airborne LIDAR Point Clouds," *Int'l Archive of Photogrammetry and Remote Sensing*, XXXV, Part B3, 2004, pp. 937-941.

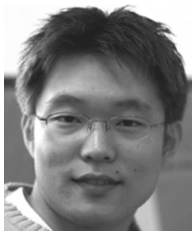
[6] W. Cho, Y.S. Jwa, H.J. Chang, and S.H. Lee, "Pseudo-grid Based Building Extraction Using Airborne Lidar Data," *Int'l Archive of Photogrammetry and Remote Sensing*, XXXV, Part B3, 2004, pp. 378-381.

[7] Working Group III/3 of ISPRS Commission III, "ISPRS Test on Extracting DEMs from Point Clouds: A Comparison of Existing Automatic Filters," http://www.itc.nl/isprswgIII-3/filtertest/index.html.

[8] M.B. Dillencourt, H. Samet, and M. Tamminen, "A General Approach to Connected-Component Labeling for Arbitrary Image Representations," *Journal of the ACM*, vol. 39, no. 2, 1992, pp. 253-280.

[9] K. Suzuki, I. Horiba, and N. Sugie, "Linear-Time Connected-Component Labeling Based on Sequential Local Operations," *Computer Vision and Image Understanding*, vol. 89, 2003, pp. 1-23.

Soo-Hee Han received the BS and MS degrees in GIS and remote sensing from Seoul National University, Korea, in 2000 and 2002. He is currently working toward a PhD degree in GIS and remote sensing at the same school.



Jeong-Ho Lee received the BS degree in GIS and remote sensing from Seoul National University, Korea, in 2002. He is currently working toward a PhD degree in GIS and remote sensing at the same school.

Ki-Yun Yu received the BS and MS degrees in civil engineering from Yonsei University in Korea, and the PhD degree in GIS from University of Wisconsin-Madison, in 1998. He was a Director in the Ministry of Construction and Transportation until 2000. He is now an Associate Professor of the School of Civil, Urban & Geosystems Engineering, Seoul National University in Korea.