J Inf Process Syst, Vol.13, No.3, pp.491~499, June 2017
https://doi.org/10.3745/JIPS.02.0061

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

A Fast Ground Segmentation Method for 3D Point Cloud

Phuong Chu*, Seoungjae Cho*, Sungdae Sim**, Kiho Kwak**, and Kyungeun Cho*

Abstract

In this study, we proposed a new approach to segment ground and nonground points gained from a 3D laser range sensor. The primary aim of this research was to provide a fast and effective method for ground segmentation. In each frame, we divide the point cloud into small groups. All threshold points and start-ground points in each group are then analyzed. To determine threshold points we depend on three features: gradient, lost threshold points, and abnormalities in the distance between the sensor and a particular threshold point. After a threshold point is determined, a start-ground point is then identified by considering the height difference between two consecutive points. All points from a start-ground point to the next threshold point are ground points. Other points are nonground. This process is then repeated until all points are labelled.

Keywords

3D Point Cloud, Ground Segmentation, Light Detection and Ranging, Start-Ground Point, Threshold Point

1. Introduction

Segmentation is important for both 2D and 3D data processing. In 2D image processing, segmentation is a pre-processing step for many tasks in [1-3]. For RGB-D sensors [4,5], the authors used a 2D image combined with deep data to segment each object. In our study, we focus on 3D point cloud data. Point cloud segmentation is important for any autonomous moving system, and ground segmentation is also part of said system. In this part, we have segmented a point cloud into two groups. The first group consists of ground points of terrain which a vehicle can traverse. The second group consists of non-ground points which vehicle cannot traverse such as trees, walls, cars, etc. If the terrain has an enormous gradient, the autonomous vehicle also cannot move across it. In this case, all points in this category are placed into the nonground group. For real-time autonomous moving systems, a fast ground segmentation method is required.

Previous studies about ground segmentation have been conducted; however the results did not match our expectations. Hernandez and Marcotegui [6] proposed a method which works well in a flat urban environment, however this method is not effective for sloping terrain. Moosmann et al. [7] proposed another method for solving the problem in non-flat urban environment, which is very simple and easy to implement. In this method, the authors built a graph from the point cloud and calculated the normal

Manuscript received September 8, 2016; first revision February 9, 2017; accepted April 10, 2017.
Corresponding Author: Kyungeun Cho (cke@dongguk.edu)
* Dept. of Multimedia Engineering, Dongguk University, Seoul, Korea (minhphuong.simtech@gmail.com, {sjcho, cke}@dongguk.edu)
**Agency for Defense Development, Daejeon, Korea ({sdsim, khkwak}@add.re.kr)

vector of each surface in the graph. Next, they defined local convexity features between pairs of neighborhood surfaces. This method is not effective when applied to bumpy terrains in mountain areas, as the point cloud may contain holes due to lost points, causing difficulties when building graphs and defining local convexity features.

In [8] and [9], the authors propose a different method for segmenting point cloud 3D data. In this method, the authors divided a point cloud dataset into smaller parts such as voxels or blocks. After processing, these smaller parts are combined to produce results, however this method is time-consuming. In [10], the authors proposed a method which is closest to our method, however the implementations are different. For each point, we must wait until obtaining enough 4 neighbor points for calculating. This method is also dependent on previous and next vertical line. Moreover, this method needs more time to build the terrain mesh.

In our method, we divide each frame of point cloud into small groups. Each group is a vertical line from sensor's position to one point in the largest circle. In the vertical line, we find all start-ground and threshold points. In practice, we often obtained one start-ground point and one threshold point. After that, we will assign label "ground" or "non-ground" for each point. The process will be repeated until we label all points.

2. Fast Ground Segmentation Algorithm

The fast ground segmentation algorithm is illustrated in Fig. 1. "Local point cloud," and is taken from the Velodyne Lidar sensor, frame by frame. Each frame contains numerous 3D points. The "Local point cloud" is processed directly in each frame but it is similar to a database. All points in each frame are in local coordinate and the original coordinate is sensor's position. We will segment each frame, and therefore do not need to convert to global coordinates. A frame data is divided into small groups. Each group is a vertical line. All points in each vertical line were processed and labelled. The details of the vertical line processing are described below.

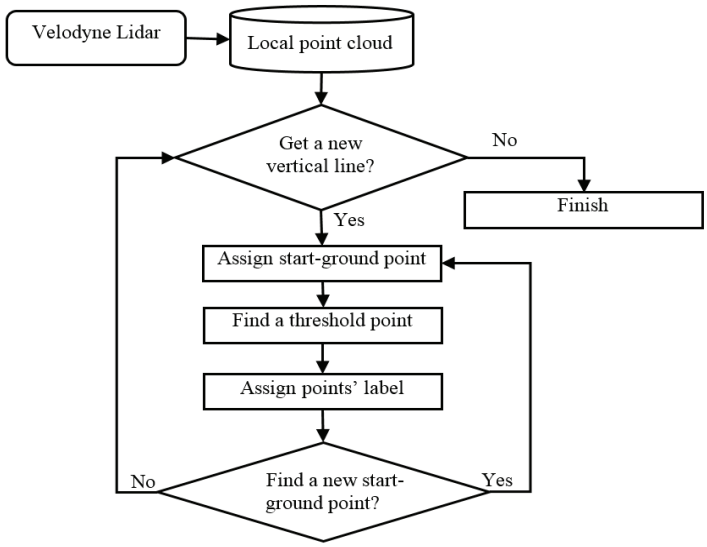

Fig. 1. Overview of the fast ground segmentation algorithm.

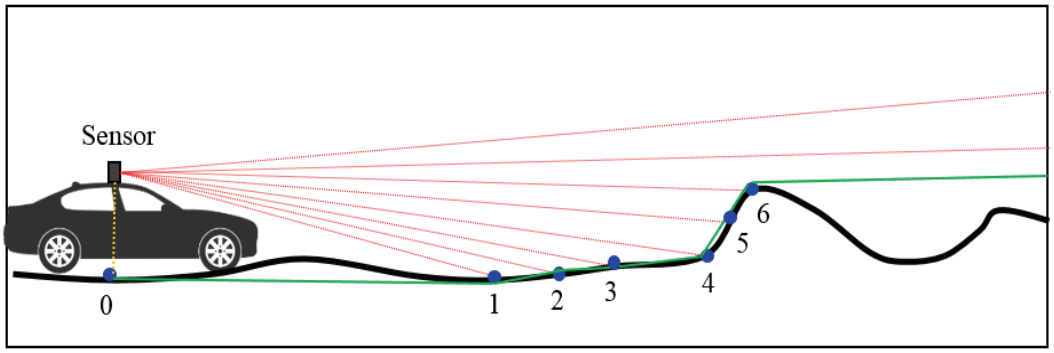

Fig. 2. A vertical line in side view.

We processed each vertical line illustrated in Fig. 2 as a green line. In theory, if we use a *n*-lines sensor, we can obtain n points in each line. We added one more point, which is a ground point the at sensor's location, making number of points *n*+1. In each vertical line, we first identified all start-ground points and threshold points. The first start-ground point is always at the location of the sensor. From this start-ground point, we then identified first threshold point by considering each pair of consecutive points. Next, whether or not a threshold point could be identified, we assigned each point a label. We also assigned a ground label from each start-ground point to the next threshold point. After each threshold point, we assigned non-ground labels to the distance between the threshold point and the next start-ground point.

We then repeated this process until all points were identified and labelled. In Fig. 2, the first start-ground point is labelled "0." We assumed that point 4 is the unique threshold point. Therefore, 1, 2, 3, and 4 are ground points and all points after point 4 are non-ground points.

2.1 Finding a Threshold Point

We considered gradient value of two consecutive points where possible. If a threshold point could not be identified, all points in the vertical line were considered ground points. As mentioned earlier, threshold points are determined by considering three features: gradient, lost threshold points, and abnormalities in the distance between the sensor and a particular threshold point.

In the first case, we calculated the gradient value α between two consecutive points using the formula below (1).

$$\alpha = \arcsin(h/d) \tag{1}$$

Here, '*h*' and '*d*' are the height difference and distance between previous and current points, respectively. We defined a maximum sloping angle of terrain which the vehicle can traverse as α_{max}. If the gradient value is larger than α_{max}, the previous point is considered a threshold point in this line. For example, in Fig. 3, C is considered a threshold point because the gradient from C to D exceeds the maximum value.

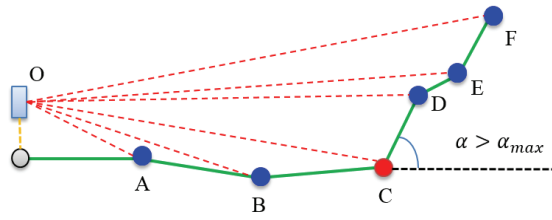

Fig. 3. Gradient which exceeds the limit value.

The second case is based on real-time observation, as a laser ray must collide with a physical object for responsive points to be identified. Using this method, we can identify all lost points in each line. If a lost point occurred between two continuous points, the previous point is considered a threshold point, even if α is smaller than α_{max}. We also propose a minimum height h_{min} to decrease noise. When h is larger than or equal to h_{min} and there is at least one lost point between previous and current points, the previous point is a threshold point. Fig. 4 shows an example of the second case, where 'B' is a threshold point because there are two lost points between B and C. In addition, the height difference between B and C is higher than the minimum height value.

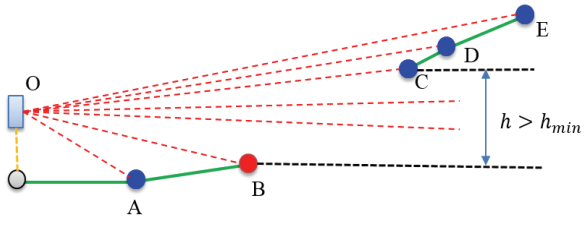

Fig. 4. Lack of points between two consecutive points.

The third case is similar to the second case. Generally, the distance from the sensor to the previous point is always shorter than the distance from the sensor to the current point. In opposite case, we also concluded that the previous point is a threshold point. This situation usually occured when the sensor moved in terrain which had many trees. In Fig. 5, point D is a threshold point because of the abnormality of the distance between the sensor and point D. In this situation, OD is greater than OE.

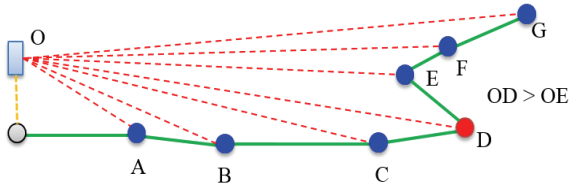

Fig. 5. Abnormality of the distance between the sensor and a point.

2.2 Finding Next Start-Ground Point

After a non-ground point, a point lower than the previous point was detected, it could be a new start-ground point, which we have called a potential point. Objects such as cars have different heights at different locations. For example, the roof of a car is higher than front and rear. Because of these

variables, if we had one point on the roof of a car and the following point on the rear, both of these are considered non-ground points. Therefore, some potential points cannot be start-ground points. For consistency, we calculated the height difference between a potential point and the nearest threshold point. If the height difference was smaller than h_{min}, the potential point was considered a new start-ground point. Otherwise, we assigned the "non-ground" label to the previous point and attempted to find a new start-ground point by considering the current and following points. In Fig. 6, B and G are threshold points. E and F are potential candidates for start-ground points. However, F is only a new start-ground point because E does not meet the second condition.

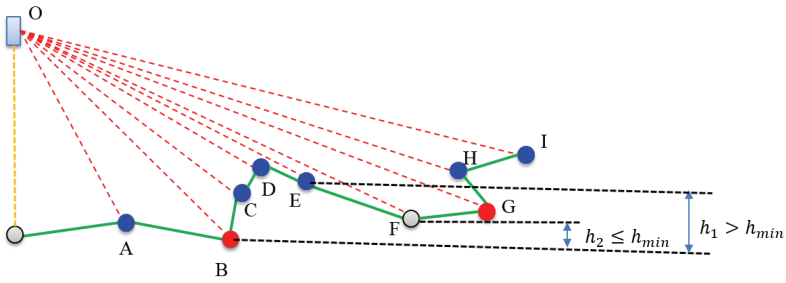

Fig. 6. Example of a next start-ground point.

3. Experiments and Analysis

For our experiments and analysis, we employed datasets obtained from a Velodyne HDL-32E sensor. We also used a sample dataset given by Velodyne LiDAR Inc. [11]. To run our algorithm, we used a PC equipped with an Intel Core i5-4690 3.5 GHz CPU and 8 GB RAM. In this experiment ground and non-ground points are not fixed. For example, if a slope is larger than 50° and the vehicle is not able to climb it, we concluded that the slope was a non-ground object. Otherwise, the slope is considered a ground object. We defined the h_{min} value to ignore the noises of the 3D range sensor. In our experiments, we chose either α_{max}=45° or h_{min}=10 cm depending on the capabilities of our vehicle and sensor. These values are not dependent on the features of the terrain. For other vehicles, different values can be chosen.

All experiments produced favorable results on both flat and non-flat terrains. Fig. 7 shows the result from one frame of data segmented at a crossroad. Figs. 8 and 9 show the results after the accumulation of many frames in a single point cloud. Red and blue points represent ground and non-ground points, respectively. The road is well segmented and defined as ground, and the trees as non-ground. In addition, Fig. 9 shows the side view of a varying terrain. The terrain contains both of flat and sloping mountain roads, however the segmentation results met our expectations.

Each frame of data contains approximately 60,000 points. We showed the processing time by frame in Fig. 10. The average processing time of our algorithm was 4.7 ms. Therefore, we have possibility to process 213 frames per second (213 Hz). Moreover, the Velodyne sensor returns only 10 frames per second (10 Hz). Therefore, our segmentation system works well in real-time as it operates 21 times faster than the sensor. We also compare the processing time of our research with previous methods as shown in Table 1 producing favorable results at higher speeds than previous methods.

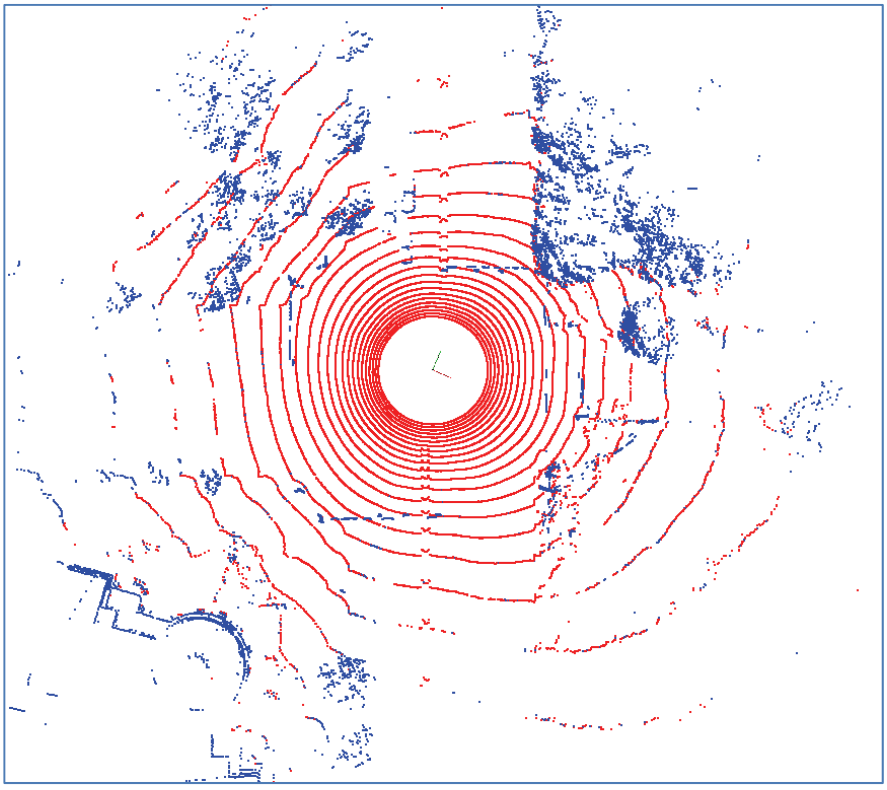

Fig. 7. Result of one frame segmentation in top view.

Fig. 8. Result of a flat terrain segmentation in perspective view.

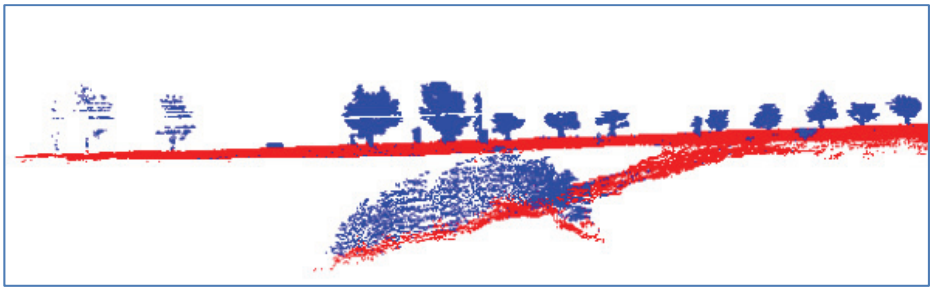

Fig. 9. Result of a sloped terrain segmentation in side view.

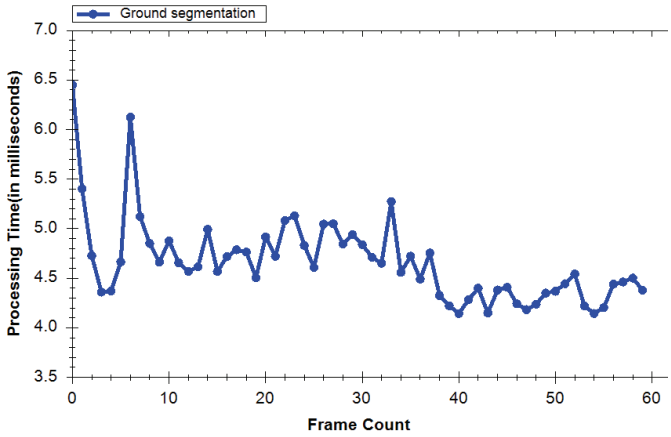

Fig. 10. Processing time by frame of our ground segmentation algorithm.

Table 1. Comparison of processing time per frame

Method	Average processing time per frame (ms)
[7]	602
[8]	19.31
Our method	4.7

4. Conclusion

In this paper, we proposed a new approach for a preprocessing step of an autonomous moving system using a Velodyne Lidar sensor. Our ground segmentation algorithm processed each frame of data using local coordinates. The core function of our algorithm is the ability to divide the point cloud in each vertical line. In addition, we considered some features of the vertical line via threshold and start-ground points. The experiments showed that the proposed algorithm achieves favorable results on data acquired in both flat and sloped environments, with an average processing time of 4.7 ms. The results are met our expectations. In future work, we will enhance the algorithm for segmentation of bumpy terrain.

Acknowledgement

This research was supported by a grant from Agency for Defense Development (No. UD150017ID).

References

[1] J. Lee, H. Chae, and K. Hong "A fainting condition detection system using thermal image cameras," *Journal of Convergence*, vol. 6, no. 3, pp. 1-15, 2015.

[2] K. M. Prabusankarlal, P. Thirumoorthy, and R. Manavalan, "Assessment of combined textural and morphological features for diagnosis of breast masses in ultrasound," *Human-centric Computing and Information Sciences*, vol. 5, no. 1, pp. 1-17, 2015.

[3] S. Lee, and E. Cha, "Style classification and visualization of art painting's genre using self-organizing maps," *Human-centric Computing and Information Sciences*, vol. 6, article no. 7, 2015.

[4] Z. Tomori, R. Gargalik, and I. Hrmo, "Active segmentation in 3D using kinect sensor," in *Proceedings of International Conference on Computer Graphics Visualization and Computer Vision*, Plzen, Czech, 2012, pp. 163-167.

[5] M. Wallenberg, M. Felsberg, P. Forssen, and B. Dellen, "Leaf segmentation using the Kinect," in *Proceedings SSBA'11 Symposium on Image Analysis*, Linkoping, Sweden, 2011.

[6] J. Hernandez, and B. Marcotegui, "Point cloud segmentation towards urban ground modeling," in *Proceedings of 2009 Urban Remote Sensing Event,* Shanghai, China, 2009, pp. 1-5.

[7] F. Moosmann, O. Pink, and C. Stiller, "Segmentation of 3D Lidar data in non-flat urban environments using a local convexity criterion," in *Proceedings of IEEE Intelligent Vehicles Symposium*, Xi'an, China, 2009, pp. 215-220.

[8] S. Cho, J. Kim, W. Ikram, K. Cho, Y. Jeong, K. Um, and S. Sim, "Sloped terrain segmentation for autonomous drive using sparse 3D point cloud," *The Scientific World Journal*, vol. 2014, article no. 582753, 2014.

[9] X. Lin and J. Zhanga, "Segmentation-based ground points detection from mobile laser scanning point cloud," in Proceedings of the International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences and 2015 International Workshop on Image and Data Fusion, Hawaii, HI, 2015, pp. 99-102.

[10] B. Douillard, J. Underwood, N. Kuntz, V. Vlaskine, A. Quadros, P. Morton, and A. Frenkel, "On the segmentation of 3D LIDAR point clouds," in *Proceedings of IEEE International Conference on Robotics and Automation*, Shanghai, China, 2011, pp. 2798-2805.

[11] Velodyne LiDAR sample data [Online]. Available: https://midas3.kitware.com/midas/community/29.

Phuong Chu http://orcid.org/0000-0002-3213-1852

He received his Bachelor Degree in Information Technology in 2011 from Le Quy Don Technical University, Hanoi, Vietnam. After that, he worked in Institute of Simulation Technology in the same university. Since September 2014, he is in the M.Eng. course at the Department of Multimedia Engineering, Dongguk University, Seoul, Korea. He has done many works mainly associated with 3D simulation applications in artificial intelligence areas and human computer interaction. Since September 2016, he is a PhD candidate in the Department of Multimedia Engineering from Dongguk University.

Seoungjae Cho http://orcid.org/0000-0003-0243-2491

He received his B.Eng. degree in Multimedia Engineering in 2012 from Dongguk University, Seoul, Korea. Since March 2012, he is in the M.Eng. and Ph.D. Eng. integrated course at the Department of Multimedia Engineering, Dongguk University, Seoul, Korea. He has done many works mainly associated with 3D simulation applications in the military and human computer interaction, artificial intelligence areas, such as remote terrain visualization and sensor simulation, brain computer interface games, robot learning, and so on. His current research interests are focused on the areas of sensor simulation applications using 3D technology and NUI (Natural User Interface) utilizing various NUI devices.

Sungdae Sim

He is a senior researcher in Agency for Defense Development, Korea. He received his B.Eng. degree in Electronic Engineering in 2004 from Kyungpook National University, Daegu, Korea. After that, he received his M.Eng. degree in Electronic Engineering in 2006 from Pohang Science and Technology Institute, Pohang, Korea. Since January 2006, he has worked at current position. His current interests are focused on sensor calibration, 3D visualization, object recognition, perceptions for autonomous vehicles.

Kiho Kwak

He a senior researcher in Agency for Defense Development, Korea. He received his B.S. and M.S. degrees from the Korea University in 1999 and 2001, respectively, and Ph.D. in ECE from the Carnegie Mellon University (CMU) in 2012. His research interests include sensor fusion, online object modeling and perception and navigation for autonomous vehicles in outdoor environment.

Kyungeun Cho http://orcid.org/0000-0003-2219-0848

She is a full professor at the Department of Multimedia Engineering at the Dongguk University in Seoul, Korea since September 2003. She received her B.Eng. degree in Computer Science in 1993, her M.Eng. and Dr. Eng. degrees in Computer Engineering in 1995 and 2001, respectively, all from Dongguk University, Seoul, Korea. During 1997–1998 she was a research assistant at the Institute for Social Medicine at the Regensburg University, Germany, and a visiting researcher at the FORWISS Institute at TU-Muenchen University, Germany. Her current research interests are focused on the areas of intelligence of robot and virtual characters and real-time computer graphics technologies. She has led a number of projects on robotics and game engines and also has published many technical papers in these areas.