3D Segmentation Method for Natural Environments based on a Geometric-Featured Voxel Map

Victoria Plaza*, Fakhr-Eddine Ababsa**, Alfonso J. Garcia-Cerezo* and Jose Antonio Gomez-Ruiz*

**University of Malaga, 29071 Malaga, Spain.*
Email: victoriaplaza, ajgarcia, janto@uma.es
***Evry University, IBISC EA 4526 France*
Email: fakhr-eddine.ababsa@univ-evry.fr

Abstract –This work proposes a new segmentation algorithm for three-dimensional dense point clouds and has been specially designed for natural environments where the ground is unstructured and may include big slopes, non-flat areas and isolated areas. This technique is based on a Geometric-Featured Voxel map (GFV) where the scene is discretized in constant size cubes or voxels which are classified in flat surface, linear or tubular structures and scattered or undefined shapes, usually corresponding to vegetation. Since this is not a point-based technique the computational cost is significantly reduced, hence it may be compatible with Real-Time applications. The ground is extracted in order to obtain more accurate results in the posterior segmentation process. The scene is split into objects and a second segmentation in regions inside each object is performed based on the voxel's geometric class. The work here evaluates the proposed algorithm in various versions and several voxel sizes and compares the results with other methods from the literature. For the segmentation evaluation the algorithms are tested on several differently challenging hand-labeled data sets using two metrics, one of which is novel.

3D Segmentation Method for Natural Environments based on a Geometric-Featured Voxel Map

Victoria Plaza*, Fakhr-Eddine Ababsa†, Alfonso J. Garcia-Cerezo* and Jose Antonio Gomez-Ruiz*
*University of Malaga, 29071 Malaga, Spain.
Email: victoriaplaza, ajgarcia, janto@uma.es
†Evry University, IBISC EA 4526 France
Email: fakhr-eddine.ababsa@univ-evry.fr

Abstract—**This work proposes a new segmentation algorithm for three-dimensional dense point clouds and has been specially designed for natural environments where the ground is unstructured and may include big slopes, non-flat areas and isolated areas. This technique is based on a Geometric-Featured Voxel map (GFV) where the scene is discretized in constant size cubes or voxels which are classified in flat surface, linear or tubular structures and scattered or undefined shapes, usually corresponding to vegetation. Since this is not a point-based technique the computational cost is significantly reduced, hence it may be compatible with Real-Time applications. The ground is extracted in order to obtain more accurate results in the posterior segmentation process. The scene is split into objects and a second segmentation in regions inside each object is performed based on the voxel's geometric class. The work here evaluates the proposed algorithm in various versions and several voxel sizes and compares the results with other methods from the literature. For the segmentation evaluation the algorithms are tested on several differently challenging hand-labeled data sets using two metrics, one of which is novel.**

I. Introduction

A segmentation method is required to isolate individual objects in a scene as a prior process to the classification or semantic recognition of them. Much effort has been made to solve the problem of segmenting in 2D images, and more recently the challenge has also been applied to the three-dimensional (3D) point clouds, as 3D sensors have become more widely used.

There are two main approaches for 3D segmentation . The first approach, called point-based techniques, is based on analyzing each point with features from its local neighborhood. Some works start the process extracting the ground by fitting it in a flat plane [1] [2] or by using a height threshold [3]. Then the non-ground points are grouped in objects or segments. These groups can be obtained depending on the distance between points by using a graph based algorithm such as Normalized Cut or Ncut [4] [3] and the algorithm of Felzenszwalb and Huttenlocher (FH) [5] over the range image [1] or over the 3D point cloud [6]. Other methods perform the segmentation based on the similarity of the normal vector computed at each point taking into account the local neighbor points. In [7] the normal at each point is computed with the information of the k-Nearest Neighbors (KNN). To reduce the normal estimation at each point, in [2] the points are first clustered if they are nearer than a distance threshold, which is called the growing region technique, then these clusters are grouped based on the similarity of the normal. In an other way, [8] the authors classify each point in ground, planar, and non-planar points through a Support Vector Machine (SVM) whose inputs are local geometric features at each point, with KNN. These features are: normal vector, height above the lowest point in the scene, or eigenvalues from the covariance matrix The planar clusters are segmented into two stages: by using a Gaussian sphere, and then refining with a distance-based clustering method, that is merging if distances are similar. The non-planar points are clustered based only on distance. The main drawback of this segmentation approach is the high memory and execution time consumption since it has to process each point in the point cloud. Therefore these methods may be incompatible with Real-Time applications.

Another set of techniques for 3D segmentation is the group-based techniques, which aim to solve the time-consumption problem of the point-based techniques by reducing the scene to a voxel model. Usually the scene is discretized in 3D cubes with constant size as a 3D grid, but these voxels can also be dynamically positioned around groups of points [9]. In [10] several algorithms have been developed for both dense and sparse pointclouds, which depends on the scan device. The work concludes that algorithms which extract the ground offer more accurate results in the subsequent segmentation process . In sparse point clouds, as acquired from a Velodyne scan, an interpolation method is performed to propagate the ground to areas without 3D information. On the contrary, for dense point clouds, as obtained with a Riegl scan, the ground is detected as the largest area with adjacent voxels with height mean or a vertical variance less than a threshold. Once the ground has been extracted, the objects are isolated by merging adjacent voxels or voxels in a distance lower than a predefined threshold. In [11] the segmentation is also with adjacent voxels, and the results are contrasted and fused with an RGB data segmentation process. In addition the ground is extracted through an elevation map. The main problem of these techniques is that they assume flat and continuous ground, as in an urban environment. Hence, big slopes or non-flat areas are not grouped in the ground. A novel idea for the voxel map is introduced in [12] called Geometric-Featured Voxel maps (GFV). In this model the voxels contain geometric information taken from a prior classification in linear or tubular structures, scattered shapes, horizontal planes, or vertical planes [13]. The ground is extracted as horizontal surface voxels with no other kind of voxels below. Horizontal surface voxels higher than a threshold over another identical kind of voxel in the same position are discarded. This way, slopes can be detected as the floor depending on the previous GFV classification, but some horizontal surfaces can be wrongly detected as ground. For instance, horizontal surface from an object may not have points below because of the shadows in the scene captured. Furthermore, another important disadvantage to this technique is that GFV classification may not be enough since the surfaces are classified in vertical and horizontal, hence it requires a very accurate threshold for slopes. All these methods have been designed for urban environments and thus ground detection is risky in unstructured environments where big slopes or non-flat areas belong to the ground.

This paper proposes a new 3D segmentation algorithm for dense point clouds and has been specially designed for natural environments. The main contribution of the work preented here is the ground segmentation method where big slopes, non-flat areas, and isolated areas which may appear in unstructured environments have been considered. Moreover, a double segmentation is implemented. The first one splits the point cloud into objects by merging adjacent voxels, since after the ground extraction the objects appear separated. Secondly, we split each object into regions with the same geometric class, by merging adjacent voxels with same class inside each object. This technique is based on Geometric-Featured Voxel maps (GFV) [13] where the scene is discretized in constant size 3D cubes or voxels which are classified in geometric classes. In this case, the classification is performed in surface, line or tubular structures and scattered or undefined shapes, usually corresponding to vegetation [14]. Since it is not a point-based technique the computational cost is significantly reduced, hence it may be compatible with Real-Time applications. Considering that the proposed ground segmentation method involves several filters or steps, several combinations of them with different orders have been tested to discern the best performance. The proposed algorithm has been tested in urban and natural environments presenting better results in the comparison with other methods from the literature in natural environments. In addition an experiment has been executed to conclude which size of voxel provides the best result. Besides a metric from the literature, a novel metric is used to quantify the accuracy in the segmentation trough a comparison between each segmented point cloud and the same one, manually labeled. The aim of this study is to provide a novel 3D segmentation method that can facilitate tasks which belong to a Search and Rescue Robot such as object recognition, traversable ground extraction and path planning in natural environments.

This paper is organized as follows. Section II describes the steps of the proposed algorithm; Section III explains the experiments and metrics to evaluate the algorithm and shows the results obtained. Finally the Section IV presents the conclusions and future work.

II. 3D Segmentation Algorithm

The input to the segmentation algorithm is not the point cloud but rather a GFV map, which is the point cloud modeled as a voxel map where each voxel keeps geometric information, as Section II-A describes. The proposed segmentation method is divided into several stages. First the ground is extracted from the scene using several filters to improve the accuracy in unstructured environments (Sec. II-B). After that, the scene is segmented first into objects and then the objects are split into regions with the same geometric classification (Sec. II-C).

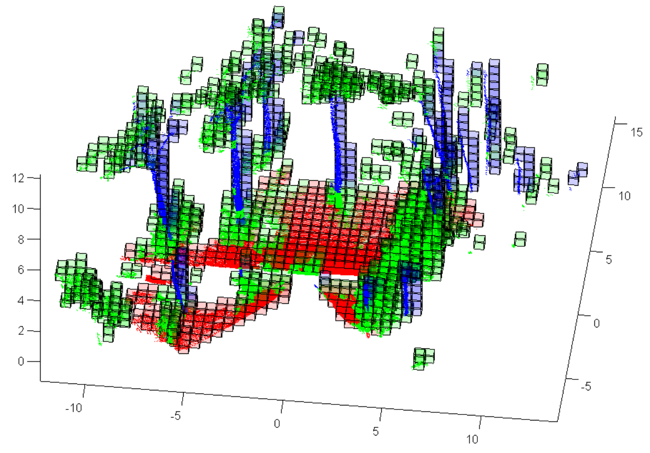

Fig. 1. Geometric-Featured Voxel Map from a natural environment. Red voxels represent flat surfaces, blue for tubular structures as trunks of the trees, and green for scattered voxels as vegetation.

A. *Geometric-Featured Voxel Map*

To reduce the computational load of dealing with each point separately, the space of data points is discretized into a regular grid of three-dimensional voxels. By default, a voxel size is 0.5 meters^3, however owing to the fact that the voxel size affects the accuracy in the segmentation process; this parameter is tested in Section III .

Once the voxel map has been obtained, the classification can be processed. The classification algorithm relies on geometric features to divide the 3D scene voxel into planar surface, linear or tubular structures and scattered regions, usually corresponding to vegetation [14]. A neuronal network has been used to classify each voxel based on the principal components from its covariance matrix. The neuronal network has been previously trained with several hand-labeled scenes which are different to the scenes used in the algorithm evaluation (Sec. III). The covariance matrix for each voxel is determined by the spatial distribution from its inner 3D points. As a result, we obtain a Geometric-Featured Voxel map where each voxel is labelled as one of these geometric classes: scatter, linear or flat surface. The Figure 1 shows a dense point cloud modelled as a GFV map. The scene shows a natural environment in Malaga captured by a low-cost laser range finder called Unolaser [15].

B. *Ground Segmentation*

Some works have tested the benefit of using the ground as a separator between objects [10]. Generally, ground segmentation methods assume the whole ground is continuous, almost flat, or the biggest object in the scene. This assumption is not true in natural environment where the ground has big slopes or rough terrain covered by grass, rocks or bushes.

Here we present a ground segmentation method specially designed for unstructured environments, this way the ground does not mean traversable ground as other methods for urban environments do. Several steps or filters compose the proposed algorithm, as the following code shows. First we consider the ground as a mesh of the lowest cubes in each position XY and then some filters discard cubes which do not belong to the ground and other filters add cubes which do.

```
Require: voxelmap
Ensure: voxelground
1: voxelground ← LowestVoxels(voxelmap)
2: voxelground ← ExtractVoxelDiscontinous(voxelground)
3: voxelground ← ExtractAreaDiscontinuous(voxelground)
4: voxelground ← AddCrest(voxelground)
5: voxelground ← AddSlope(voxelground)
```

1) Lowest voxel in each sole cell: Considering the 2D grid from the voxel map sole, this is the plane XY, each cell may involve one or more voxels, or none, with different heights (coordinate Z) but the same XY position. Therefore we assume the lowest cube in each cell from that grid belongs to the ground. This is the main step and after it all the filters can be processed in different orders.

2) Extract voxels with vertical discontinuity: Assuming the ground can contain big slopes but not completely vertical ones, because vertical slopes belong to an object, then two adjacent ground cubes must have adjacent heights. That is, knowing that a cube A belongs to the ground, then a cube B adjacent to A in the plane XY, cannot belong to the ground except when it has the same height as A, or its upper or lower cube height. Ground cubes with a height difference bigger than a cube size will be discarded. With this filter we can discard voxels which are much further up than their neighbors (see Figure 2).

3) Extract voxels in isolated areas with vertical discontinuity: Some ground cubes are grouped in an area not adjacent to the rest of it, so the vertical steps have to be filtered in another way. Ground voxels with adjacency are grouped in areas and we assume the area or areas containing a voxel with the lowest height are true ground areas, so then we test the areas left. In each area, the height step is estimated as the height difference between its lowest cube and the nearest voxel in a true ground area. As in the previous filter, if this height difference is not adjacent, it means, the height step is bigger than a voxel size, so the area is discarded from the ground areas set (see Figure 2).

4) Add voxels in a terrain crest: So far, the ground has been composed by one cube per cell maximum, but the relief of the ground may require two cubes per cell. When a ground voxel has an upper voxel with i) no voxel on top and ii) a height variance from its inner points less than a fixed threshold, then this upper voxel is included as a ground voxel (see Figure 3).

5) Add voxels in a slope: Each slope between two adjacent ground voxels requires a new voxel on top of the lower voxel in the slope, except when the slope is perfect and passes through the edges of both voxels which is the ideal case and highly improbable. Generally, we look for two voxels A and B which belong to the current ground voxel list and they have some restrictions: i) ground voxels A and B are adjacent voxels in plane XY, ii) voxel B is the same height as A's upper voxel height and iii) A's upper voxel exists and does not belong to the ground. Therefore a slope has been recognized, and the voxel on top of A is added as ground (see Figure 3).

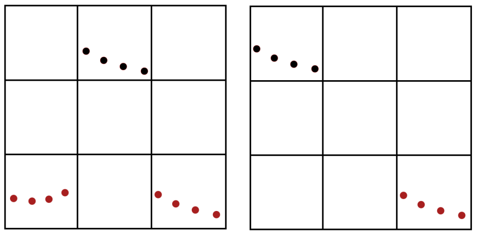

Fig. 2. Point clouds silhouettes where the extracting filters are required. In red the lowest point at each cell which belong to the ground, in black the lowest point at each cell which should be extracted from the ground segment. In the picture on the left there is a voxel with a vertical discontinuity with respect to its adjacent voxel in XY. In the picture on the right there is an isolated area with vertical discontinuity.

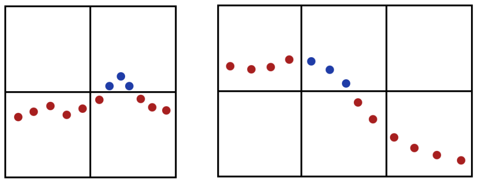

Fig. 3. Ground silhouettes where the adding filters are required. In red the lowest point at each cell, in blue other ground points. On the left, a terrain crest, on the right a slope.

6) Add voxels in a slope, recursive: As the previous filter but considering in the process the new voxels added to the ground since they can also obey the restrictions and belong to a slope in the ground.

C. Object Segmentation

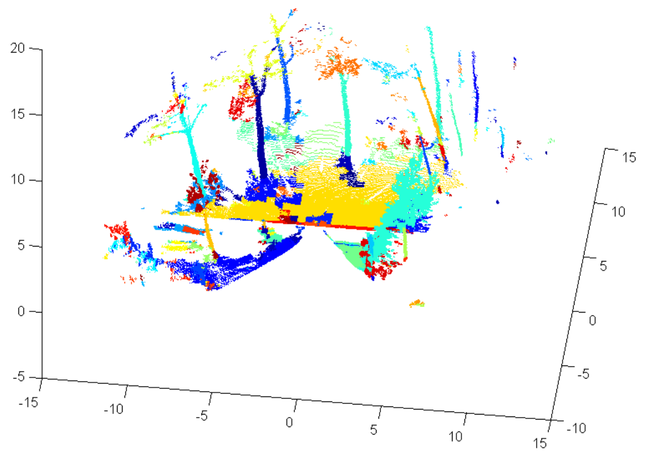

Fig. 4. Scene with objects segmented into regions with the same geometric class: surface, linear, scatter. Random color for each segment.

Once the ground has been extracted, the object partition is found by clustering together adjacent voxels [10], see Figure 5. Each object is also split into regions. These regions are obtained by merging adjacent voxels, based on the same geometric class, see Figure 4. For example a tree object would be split into two regions at least: i) a linear region relative to the trunk and branches and ii) a scattered region relative to the tree crown. Once we have the regions from each object, we can compute several statistical features as: width, height, depth, centroid position and distance from the ground. These features may be used for future classification of the objects. For instance, an object with a scattered region at the top and a long linear region below, just over the ground, could be labeled as a tree.

III. Evaluation

To evaluate the proposed segmentation method we have used a set of three differently challenging scenes which are called Urban, Dense and Slope scenes, respectively. For the comparison with previous work, one of the scenes used is a very large point cloud with many different urban elements from the University of Sydney campus [10] captured by a Riegl scan. For this experiment we use a part of the scene with 41 big objects including buildings, trees, cars, bushes, fences and light poles. The other two point clouds show a natural environment in Malaga captured by a low-cost laser range finder called Unolaser [15]. One of them represents a dense forest with many trees and bushes over the ground and the other contains a steep slope and just a tree and a bush. Noisy points from sparse areas will not be taken into account in the experiment since they can not be manually identified as an object.

Two metrics have been used to quantify the agreement between each object automatically segmented with the proposed method and the same object in the hand segmented scene. The evaluation method is iterated for all the hand-labeled objects sorted in descending order by the number of inner points. At each hand-labeled object, we extract the same points in the scene resulting in our algorithm. These points can belong to more than one object because of the oversegmentation, so the largest object is considered the match, and all other partitions are considered errors. Furthermore, to quantify the error from undersegmentation, once an object has been counted as a match, it is considered as an error in any subsequent match attempts. This first metric was proposed in [10] but it does not consider the ground as an special object, and the undersegmentation is not quantified enough.

The second metric that we propose considers the undersegmentation as a bigger penalty. When a hand-labeled object A matches an automatically segmented object A*, we count the points of A* which belong to A and the points of A* which are outside of A. In the case of a significant undersegmentation, A only contains points from A* and no other objects, but A* contains points of A and other objects B* and C*. In the first metric the points of A* outside of A, will be considered errors in the rest of the evaluation. In this way, in the evaluation of B and C, the point score will be 0. But, sometimes B* and C* do not correspond to objects in the hand-labeled scene, so this penalty will never be applied. For example, B* and C* may be areas from the ground which have been wrongly segmented as new objects. In addition, if exist A, B, an C with the same size in number of points, A will score 100%, B and C 0%. However, the strict score should be 0 in all three cases. In the second metric, the evaluation is very similar but a 0% score is assigned if A* has more points outside of A than inside it. Then, the second metric will always be smaller than or equal to the first one, but more accurate.

In our experiment, each evaluation provides several data in percent:

- (G) the point score in the ground segmentation, the number of points well matched in the ground with respect to the total number which belong to the hand-labeled segmented ground, in percent.
- (F) shows the false positive points, the number of points outside the hand-labeled ground which have been wrongly segmented in the ground with respect to the total number of ground points.
- (O) the point score in the objects left, with the first metric.
- (N) the point score in the objects left, with the novel proposed metric.
- (T) is the time of the ground and object segmentation, in seconds.

Table I shows the results of the evaluation of the scenes with different ground extraction methods with a voxel size of 50 cm^3. These methods are the combination of the filters of the ground extraction stage in different order. These filters were numbered from 1 to 6, and the order is shown in the first column. Number 7 represents the method with the best results proposed in [10] called Cluster-All. For this method, we have used the neighborhood magnitude of 1, which means we consider neighbors only those voxels which are touching, adjacent, as our proposed method. In addition the variance and height thresholds for the ground required by this method were extracted from the Urban hand-labeled scene.

The implementation of these methods has been done in Matlab, and the experiments were executed on a computer with a i7 processor with a clock frequency of 2.4 GHz and 8GB in RAM. The Urban scene presents the best results, almost perfect ground and object segmentation in all methods, but the computational times are large because of the large number of points. The Dense scene presents very good results in the ground extraction, but between 30 and 50% of point score in the object segmentation. This is because the bushes are partitioned together with the adjacent trees, and several trees are joined by their crowns (see Figure 5). In fact, it does not present a problem for the future recognition of the scene for a navigation system because the second segmentation in regions with the same geometric class shows the parts of the object which are solid obstacles (linear or tubular class) or traversable such as bushes on the ground (scatter class). Finally the most risky scene is the Slope one where the ground extraction has scores of between 50 and 73% in most of the methods, except the last three and this produces poor object segmentation. In addition, this scene shows the lack of accuracy in the first metric for the object segmentation. Figure 6 shows the Slope scene segmented with the method 12345. The main object is the tree which has been wrongly segmented from the ground, from an area of the ground which is also badly segmented. The first metric provides a score of 61% and the second metric 0.6% which is more accurate.

To sum up, there is no method with the best score in all the scenes but when analyzed, the mean of the novel proposed metric (N) in the three scenes shown in the last column (Mean), the best results are provided by the method which combines only filters 1, 6 and 3 in the ground extraction stage. These correspond to the sequence of the filters Lowest voxel in each

TABLE I. SEGMENTATION RESULTS WITH DIFFERENT GROUND EXTRACTION METHODS

Filter	Slope Scene					Dense Scene					Urban Scene					Mean[%]
	G[%]	F[%]	O[%]	N[%]	T[s]	G[%]	F[%]	O[%]	N[%]	T[s]	G[%]	F[%]	O[%]	N[%]	T[s]	
1234	50,8	0,0	61,5	0,6	4,0	99,3	4,8	61,2	36,3	6,1	100,0	0,1	94,7	94,7	85,3	43,9
12345	66,0	0,0	61,5	0,6	3,5	99,9	25,0	50,6	32,2	4,3	100,0	2,4	93,4	93,4	113,9	42,1
12346	69,8	1,4	32,4	0,6	2,7	99,9	62,6	45,9	36,4	4,5	100,0	3,6	92,7	92,7	91,6	43,2
12354	65,3	0,0	61,5	0,6	2,3	99,7	18,6	61,0	54,4	3,0	100,0	2,2	93,4	93,4	97,2	49,5
1236	69,3	1,4	32,4	0,6	2,7	99,6	51,5	50,3	40,8	1,5	99,6	3,3	92,8	92,8	100,6	44,7
12364	69,8	1,7	26,5	0,6	2,5	99,7	55,6	47,9	38,4	5,8	100,0	3,3	92,8	92,8	100,3	43,9
1423	50,4	0,0	61,5	0,6	2,6	99,3	4,7	61,0	36,3	6,1	99,9	0,1	94,7	94,7	102,1	43,9
14523	69,8	0,0	61,5	0,6	4,3	99,9	28,3	48,6	30,4	4,2	99,9	2,4	93,5	93,5	112,3	41,5
14623	73,0	0,0	61,5	0,6	2,7	99,9	64,8	44,8	35,3	6,8	99,9	3,4	93,0	93,0	113,6	43,0
15423	65,4	0,0	61,5	0,6	4,7	99,7	21,3	59,1	52,4	6,0	99,9	2,1	93,5	93,5	101,5	48,9
16230	72,6	0,0	61,5	0,6	3,5	99,6	54,4	48,4	38,9	3,8	99,6	3,2	93,0	93,0	98,5	44,2
16423	72,9	0,0	61,5	0,6	2,4	99,7	56,7	47,0	37,5	3,6	99,9	3,2	93,0	93,0	84,8	43,7
1	54,6	0,0	61,5	0,6	2,2	98,4	5,5	55,7	30,8	4,5	88,5	2,1	80,4	70,5	109,1	34,0
16	98,4	2,8	38,2	38,2	1,7	99,6	75,5	36,1	26,8	2,8	99,6	6,3	90,0	90,0	125,5	51,7
163	98,4	2,8	38,2	38,2	3,5	99,6	73,0	36,9	27,1	4,1	99,6	3,8	93,1	93,1	112,0	52,8
7	85,5	4,6	61,5	0,6	4,3	48,9	4,3	57,3	32,1	4,8	100,0	0,6	94,3	94,3	117,3	42,3

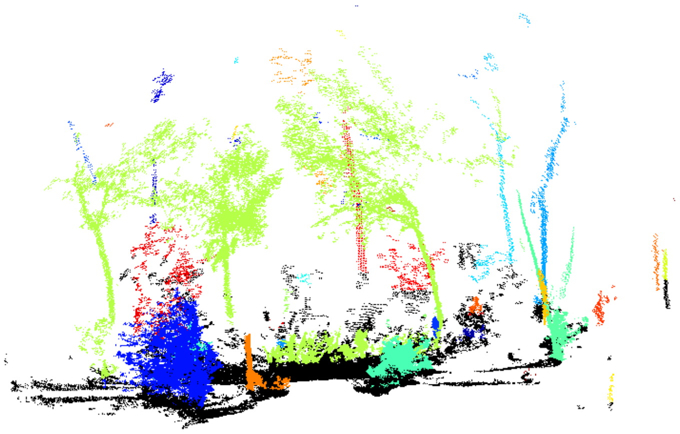
Fig. 5. Dense scene segmented with the method 12354. Black points are segmented as ground, other random colors for the segmented objects. The main three trees are joined by their crowns and some bush.

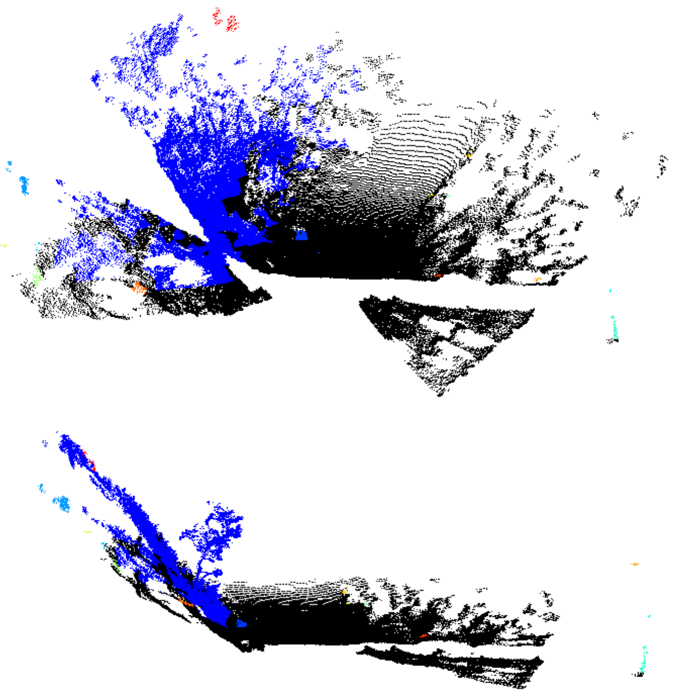
Fig. 6. Slope scene wrongly segmented. The bottom image shows the same scene from the side. Black points are segmented as ground, other random colors for the segmented objects.

sole cell, Add voxels in a slope -recursive version- and Extract voxels in isolated areas with vertical discontinuity. When the extracting filters (2 or 3) are applied before the adding filters (4, 5 or 6), some voxels required for the interpolation in a slope ground are removed. However, in scenes where the ground is almost flat and continuous, the extracting filters applied in the first place provide better time consumptions because the folowing filters will process less data, and any accuracy penalty is shown. Our method obtains a much better result than others in the literature especially in natural environments with big slopes.

In another experiment different voxel sizes are tested with the algorithm proposed 1-6-3. A too small size of voxel produces sparser data, the number of empty cubes which connect surfaces from the same object increases, then objects become over-segmentated. On the contrary, too big voxels can create adjacency in voxels which really are in different objects. This could be an advantage in sparse point clouds where two points are separated but their voxels can offer a direct adjacency, hence the segmentation process will merge these points in the same object. In dense point clouds the points are closer and the optimal voxel size will be the one which is as big as possible to further simplify the point cloud but at the same time small enough to keep the meaningful features of the scene, as shapes.

Figure 7 shows the ground and object segmentation scores for Slope and Dense scenes with the novel metric N. When the voxel size increases, the ground score also increases however the object score decreases. In conclusion, more accurate segmentation results are obtained with lower voxel sizes. Furthermore, we have analyzed the performance of the algorithm in terms of time and object score per unit of time. Figure 8 presents the decrease of the execution time as the voxel size increases, as it would be assumed that the point cloud is becoming simpler in a smaller number of voxels. However, since the available time is limited in many applications, it is useful to analyze the performance in terms of the segmentation score per each unit of the executed time. The rate of score per time is based on the result of the Point Score in the object

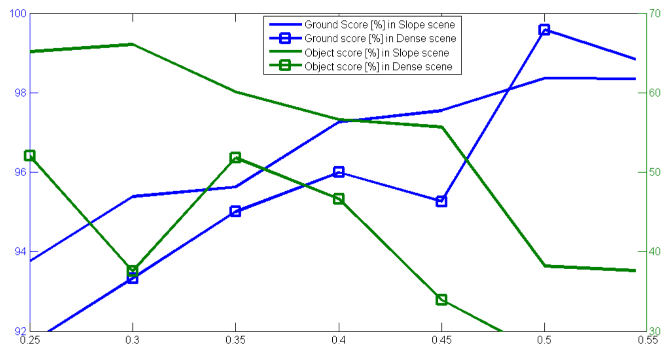

Fig. 7. Ground and Object point scores, in percent, for the data set Slope and Dense with different voxel size. The ground scores are in the blue scale, in the left vertical axis, and the object scores are in the green scale, in the right vertical axis. X axis is the size of the voxel used, in m^3

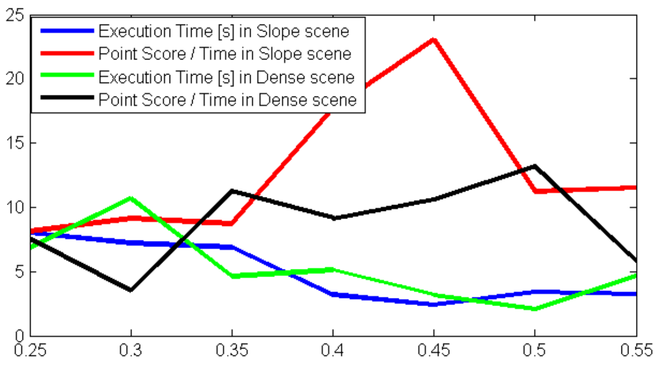

Fig. 8. Execution times in seconds and rates of segmentation scores per unit of time with different voxel sizes. Blue and green lines are the execution times in Slope and Dense scene respectively, and red and black lines are the score per time rate for these scenes, respectively. X axis is the size of the voxel used, in m^3

segmentation with the novel metric N in percent, per each unit of the execution time in seconds. The figure shows that the optimal performance is obtained with a voxel size between 0.45 and 0.5 m^3.

IV. Conclusion

This paper has proposed a new method for 3D segmentation, specially designed for unstructured environments where the ground is not flat or continuous. The method is based on a Geometric-Featured Voxel map where the dense point cloud is discretized in a 3D grid and each voxel is classified as flat surface, linear or tubular structures, or scattered area. Since it is not a point-based technique the computational cost has been significantly reduced, hence it may be compatible with Real-Time applications such as autonomous navigation for outdoor robots. Several filters has been proposed and tested for the ground extraction to obtain more accurate results in the subsequent segmentation process. The segmentation is performed in two steps, one in objects with adjacent voxels and a second and refined one, splitting each object into regions with adjacent voxels and the same geometric class. The evaluation process concludes that the proposed algorithm obtains better results than others from the literature in natural environments with rough ground. With this double-segmentation method the traversable ground can be extracted to facilitate the path planning. Solid obstacles can be labeled in linear or surface regions distinct from the ground such as trunks, and traversable objects can be labeled in short scattered regions over the ground such as bushes. In addition several geometric and statistical features can be extracted from both objects and regions in order to classify objects automatically and in the recognition of the scene.

Acknowledgment

This work was partially supported by the Spanish CICYT project DPI 2011-22443.

References

[1] X. Zhu, H. Zhao, Y. Liu, Y. Zhao, and H. Zha, “Segmentation and classification of range image from an intelligent vehicle in urban environment,” in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, Oct 2010, pp. 1457–1462.

[2] Y. Zhou, Y. Yu, G. Lu, and S. Du, “Super-segments based classification of 3d urban street scenes,” *International Journal of Advanced Robotic Systems*, vol. 9, 2012.

[3] Y. Yu, J. Li, H. Guan, C. Wang, and J. Yu, “Semiautomated extraction of street light poles from mobile lidar point-clouds,” *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 53, no. 3, pp. 1374–1386, March 2015.

[4] J. Shi and J. Malik, “Normalized cuts and image segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[5] P. F. Felzenszwalb and D. P. Huttenlocher, “Efficient graph-based image segmentation,” *Int. J. Comput. Vision*, vol. 59, no. 2, pp. 167–181, Sep. 2004.

[6] M.-C. Sima and A. Nchter, “An extension of the felzenszwalb-huttenlocher segmentation to 3d point clouds,” pp. 878 302–878 302–6, 2013.

[7] T. Rabbani, F. A. van den Heuvel, and G. Vosselmann, “Segmentation of point clouds using smoothness constraint,” in *IEVM06*.

[8] W. Hao and Y. Wang, “Classification-based scene modeling for urban point clouds,” *Optical Engineering*, vol. 53, no. 3, 2014.

[9] D. Habermann, A. Hata, D. Wolf, and F. Osorio, “Artificial neural nets object recognition for 3d point clouds,” in *Intelligent Systems (BRACIS), 2013 Brazilian Conference on*, Oct 2013, pp. 101–106.

[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 *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, May 2011, pp. 2798–2805.

[11] B. Douillard, A. Brooks, and F. Ramos, “A 3d laser and vision based classifier,” in *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2009 5th International Conference on*, Dec 2009, pp. 295–300.

[12] B. Seo and M. Chung, “Traversable ground detection based on geometric-featured voxel map,” in *FCV 2013 - Proceedings of the 19th Korea-Japan Joint Workshop on Frontiers of Computer Vision*, 2013, pp. 31–35.

[13] Y. Choe, I. Shim, and M. Chung, “Geometric-featured voxel maps for 3D mapping in urban environments,” in *9th IEEE International Symposium on Safety, Security, and Rescue Robotics, SSRR 2011*, 2011, pp. 110–115.

[14] J. Lalonde, N. Vandapel, D. Huber, and M. Hebert, “Natural terrain classification using three-dimensional ladar data for ground robot mobility,” *Journal of Field Robotics*, vol. 23, no. 10, pp. 839–861, 2006.

[15] J. Morales, J. Martinez, A. Mandow, A. Pequeno-Boter, and A. Garcia-Cerezo, “Design and development of a fast and precise low-cost 3d laser rangefinder,” in *Mechatronics (ICM), 2011 IEEE International Conference on*, April 2011, pp. 621–626.