IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 37, No. 6, December, 2020, pp. 1019-1027
Journal homepage: http://iieta.org/journals/ts

Clustering-Based Plane Refitting of Non-planar Patches for Voxel-Based 3D Point Cloud Segmentation Using K-Means Clustering

Ali Saglam[1], Hasan B. Makineci[2], Ömer K. Baykan[1], Nurdan Akhan Baykan[1*]

[1] Department of Computer Engineering, Konya Technical University, Konya 42075, Turkey
[2] Department of Geomatics Engineering, Konya Technical University, Konya 42075, Turkey

Corresponding Author Email: nbaykan@ktun.edu.tr

https://doi.org/10.18280/ts.370614

Received: 23 June 2020
Accepted: 25 November 2020

Keywords:
plane fitting, plane refitting, point cloud segmentation, plane clustering, k-means clustering, standard deviation thresholding

ABSTRACT

Point cloud processing is a struggled field because the points in the clouds are three-dimensional and irregular distributed signals. For this reason, the points in the point clouds are mostly sampled into regularly distributed voxels in the literature. Voxelization as a pretreatment significantly accelerates the process of segmenting surfaces. The geometric cues such as plane directions (normals) in the voxels are mostly used to segment the local surfaces. However, the sampling process may include a non-planar point group (patch), which is mostly on the edges and corners, in a voxel. These voxels can cause misleading the segmentation process. In this paper, we separate the non-planar patches into planar sub-patches using k-means clustering. The largest one among the planar sub-patches replaces the normal and barycenter properties of the voxel with those of itself. We have tested this process in a successful point cloud segmentation method and measure the effects of the proposed method on two point cloud segmentation datasets (Mosque and Train Station). The method increases the accuracy success of the Mosque dataset from 83.84% to 87.86% and that of the Train Station dataset from 85.36% to 87.07%.

1. INTRODUCTION

3D point clouds are unorganized data whose elements (points) are spatially located in three-dimensional space [1, 2]. These data are obtained from the surfaces of real-life environments, structures, and objects via laser scanners. Meaningful information at a high level can be extracted from the cluttered surface points by processing the data with effective methods. Processing of the point clouds is a subject of interest in the fields of computer science, photogrammetry, remote sensing, architecture, archaeology, and robotic. Through the processing of point clouds; objects, structures, and environments can be modeled in digital environments and, many real-life problems can be solved automatically [3-10].

The surface normal of a point group signifies the inclination of their fitting plane in the 3D spatial space [11]. The local plane inclinations (represented by local surface normals) and the tangent vector of the estimated common plane of two adjacent point groups (represented by the vectors between the barycenters of the adjacent local point groups) are some geometric features used as the basis for the segmentation process through many methods in the literature [12, 13]. These local features, which compose the surface gradients, are estimated by using neighboring point groups. In the literature, this neighborhood is obtained generally via two approaches which are computing the nearest neighbors for each point and grouping the points into cubic volumes (voxels) of a regular grid structure (voxelization) [14]. The voxelization approach has some advantages [15]. Firstly, the local point groups are determined faster than the nearest neighboring approach, and reaching the nearby groups is easy due to the regular indexing of the voxels. Besides these advantages, due to the noise and dense points in voxels are suppressed, the data to be evaluated in the segmentation process is reduced by avoiding paying attention to similar points one by one in very dense regions. The octree organized hierarchical dividing is mostly used voxelization technique in the literature because of the low memory usage and the indexing facilities [14, 16].

The voxelization process may come with some undesirable situations such as the voxels that fall on the points at the edges and corners of objects and structures. These voxels have not a planar feature, and thus the point groups (patches) in these voxels are named as “non-planar patches” in this paper. A non-planar patch comprises more than one small planar patch that belongs to a surface. In a voxel-based region growing point cloud segmentation process, these patches may cause a growing region to switch to another surface that shows different plane inclination. For this reason, the non-planar patches are divided into small planar patches with the k-means clustering algorithm. The normal and barycenter features of the voxel are replaced with the features of the largest small planar patch obtained after the clustering process. This process ensures the non-planar patch belongs to a surface more precisely. Before testing the non-planar patches fitted to new planes in a voxel-based segmentation algorithm, firstly the voxel-based point cloud segmentation algorithms are reviewed in the following paragraphs.

A non-planar local point group can significantly affect the segmentation of a large and important segment in a voxel-based segmentation process. In this study, a refinement method that is named “refitting” is proposed as an intermediate process for voxel-based point cloud segmentation methods. This method firstly detects the non-planar local surfaces on the edges and corners and then, clusters these surfaces into planar

sub-surfaces with a clustering method. In our experiments, this intermediate-process saves the boundaries of some important segments and increases the segmentation successes on the dataset used in the experiments. On the other hand, this process extends the duration of the segmentation, but not much compared to the normal duration. The extended time depends on the number of non-planar patches in the data and clustering sensitivity.

In many methods in the literature, the points in the clouds are voxelized as a pre-processing stage. Wang and Tseng [17] divide the point cloud by using the octree structure until each leaf node of the octree has plane features in their method. The planar point groups are seen as the vertices and, the connections between the adjacent groups are weighted by using the angles of normal vectors of the vertices. The weights that are higher than the specified criterion is cut. Vo et al. [14] improved a voxel-based region growing algorithm by using the surface normal in the weight measurements of the connections between adjacent voxels. Bassier et al. [18] proposed another voxel-based region growing algorithm by using the surface normal and the RGB color values in two separate weight measurements. Xu et al. [19] improved edge weight measurements composed with the surface normals, the vector between the barycenters, and the spatial distances of the voxels in their algorithm "Voxel and Graph-based Segmentation" (VGS). Saglam et al. [20] proposed a voxel-based segmentation method that uses only the normal and barycenter features of the voxels in their algorithm "Boundary Constrained Voxel Segmentation" (BCVS).

The supervoxel-based approaches are also prevalent in the literature. A supervoxel consists of spatially close voxels that are similar to each other according to some specified features. Papon et al. [21] used the CIELAB color value differences between the adjacent voxels as the weight value of the connections to compose the super voxels in the first level segmentation of their method "Voxel Cloud Connectivity Segmentation" (VCCS). In the second level segmentation, they merged the supervoxels using the spatial coordinates, color values, and the 33 elements of "Fast Point Feature Histograms" (FPFH) defined in Rusu's study [22]. Stein et al. [23] extended the VCCS method in their method "Local Convex Connected Patches" (LCCP). They used the surface normals in the weight values when composing the supervoxels. In the second level segmentation, they used a convexity criterion to merge the supervoxels. Zhu et al. [24] composed the supervoxels by using the surface normals, the barycenters, and the RGB color values in their weight measurement. Verdoja et al. [25] used the surface normals and the vectors between the barycenters to compose the supervoxels. Xu et al. [19] improved the VGS algorithm by using the supervoxels in their algorithm "Supervoxel- and Graph-based Segmentation" (SVGS).

In the literature, there are also some refinement methods. Vo et al. refine the points in the boundary voxels after the segmentation process. This method accomplishes the refinement at the end of the segmentation. Therefore, their refinement affects segmentation success slightly [14]. Li and Sun [26] refine the points to the nodes that represent the supervoxels by resampling the points iteratively with the k-NN algorithm. Jung et al. [27] fit the powerlines on some specific datasets using a re-clustering approach as refinement. Poux et al. refine the non-planar point groups after the segmentation process in their study [14, 28].

In this paper, the BCVS algorithm is used to test the proposed refitting method as the test segmentation method because of its success and speed of operation and needing only one parameter in addition to the parameter of voxel size [20].

In Section 2, the surface normals of the patches and the BCVS segmentation method are mentioned as preliminaries. In Section 3, the proposed refitting method for non-planar patched is explained. In Section 4, the proposed method is tested on two point cloud segmentation datasets using the BCVS algorithm, and the segmentation success as without refitting and with refitting are compared based on accuracy values and visual outputs.

2. PRELIMINARIES

In this section, some preliminary information about the surface normals of patches and the inherent voxel-based segmentation method used the experiments is mentioned. To understand the proposed plane refitting method, understanding of the geometrical expressions of local surface structures of points and the BCVS method is an especially crucial issue.

2.1 Surface normals of the patches

The local surface normal of the patch in a voxel can be estimated by utilizing the Principle Component Analysis (PCA) [29, 30]. The PCA method estimates the principle distribution directions (eigenvectors) and the variances in those directions (eigenvalues) of the data in a multi-dimensional space [3]. The PCA procedure extracts three eigenvectors e_1, e_2 and e_3 and three eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3$ (corresponding to the eigenvectors in the same order) from a point group in a 3D space. The normal vector n is represented by the eigenvector e_3 corresponding to the smallest eigenvalue λ_3. If the point group constitutes a planar structure, its normal vector is perpendicular to the plane surface as seen in Figure 1. As seen in the figure, e_1 is the orthogonal direction in which the points spread with the highest variation and e_2 and e_3 is the other orthogonal directions in which the points spread with the second and third highest variations and perpendicular to e_1.

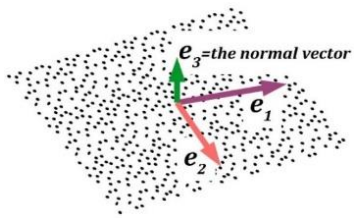

Figure 1. The normal vector among the PCA eigenvectors

Although the normals of two surfaces are parallel to each other, they can indicate the opposite directions. In order to rotate the normals onto the same aspect, an origin point and the barycenters of the voxels can be used [3, 20, 23, 25, 31]. If the angular difference between the unit normal vector of a voxel and the vector from the origin to the barycenter of the points in the voxel is higher than the angle 90°, the normal of the voxel is inverted, or vice versa. In this way, the angular differences between the normals can be up to 90°, because it is important that the inclinations of the vectors, not their directions as seen in Figure 2. If two vectors are parallel to each other but in opposite directions, nevertheless, the inclinational difference of the two vectors must be zero.

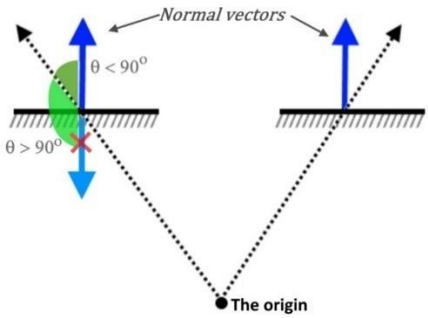

Figure 2. Setting the normal direction according to an origin

Another situation in the normal calculation is the non-surface voxels that do not have 3D surface properties. The patch in these voxels may exist in two forms. One is the patches that include less than three points [14, 32], and the other one is the patches that their points are sequenced linearly through a line [20]. Linear sequenced points can be determined by using the rate of λ_2 to λ_1 [33]. These non-surface voxels are marked as non-surface in implementation because weight values of the connections to these voxels from their adjacency voxels are computed in a different way as can be seen in the following subsection.

2.2 The segmentation algorithm used in the experiments

The BCVS algorithm [20] that is used to test the impact of the proposed intermediate process on the voxel-based segmentation process consists of some stages after the voxelization and normal estimation processes. The first stage is determining the connections between adjacent voxels, the second is weighting the connections according to the local geometric features, and the third is merging the voxels as long as the merging criteria are met until to create final segments.

The connection determining process can be accomplished using the "Connected Component Labeling" (CCL) method [34]. The adjacent leaf voxels of all leaf voxels are determined by usually 26 neighborhood (3×3×3 frame by centering the query voxel) in our implementation like many implementations in the literature.

To weight the connection between two adjacent voxels, the normals (n_i and n_j) of the voxels and the barycenters (X_i and X_j) of the points in the voxels are used according to the source study [20]. One of the values, which is used in the weight measure, is the angle θ_{ij} between the normals n_i and n_j. The angle can be computed by Eq. (1).

$$\theta = \cos^{-1}(n_i \cdot n_j) \tag{1}$$

The other value used in the weight measure is the average of the angles α_i and α_j between the patches and the vector d_{ij} between X_i and X_j respectively as illustrated in Figure 3 (a). To obtain the two angles, firstly the angles β_i and β_j between the normal vectors and d_{ij} is computed (if any of these angles is largest than 90°, the angle is replaced with its supplementary to 180° shown in Figure 3 (b)) and then, the complementary angles of them to 90° give the angles α_i and α_j. After the angles θ_{ij}, α_i and α_j are computed, the weight value w_{ij} is computed according to Eq. (2).

$$w_{ij} = \min\left(\theta_{ij}, \frac{\alpha_i + \alpha_j}{2}\right) \tag{2}$$

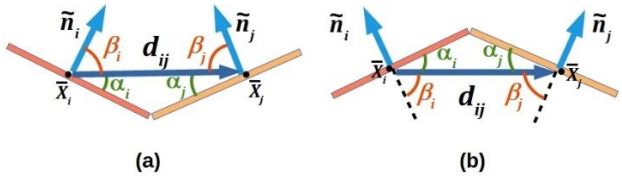

Figure 3. Angles between the vectors

Since the normals of the non-surface voxels cannot be computed, the weight value of the connections to these voxels can be computed using the angles α_i and α_j. Although they have not a normal vector, they have barycenter. In the weighting the connection between the voxels V_i and V_j, if the V_i is non-surface, w_{ij} is α_j, or vice versa. If both of them are non-surface, the connection between them is removed.

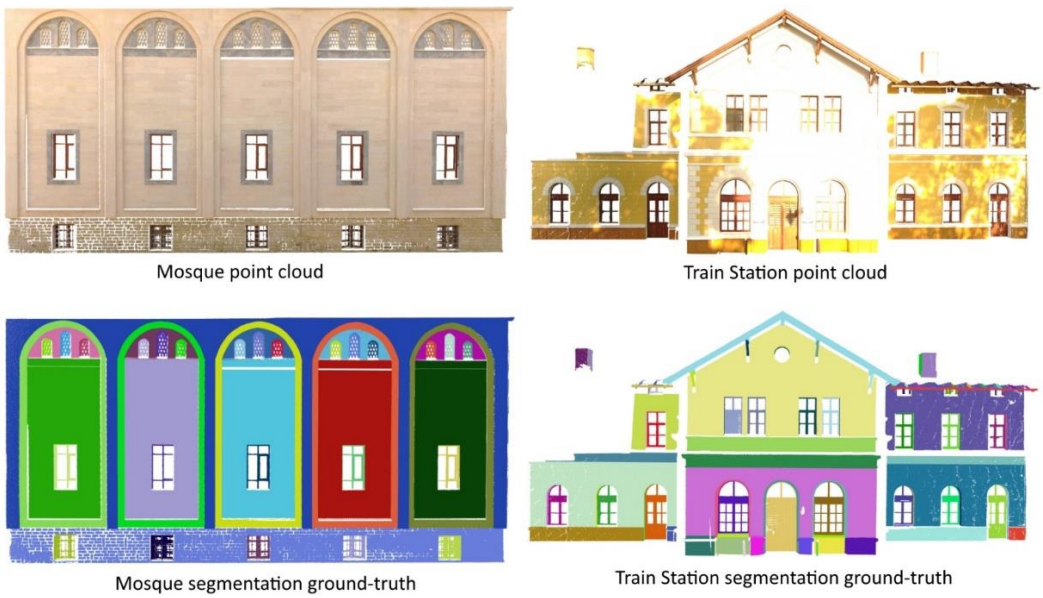

Figure 4. The datasets used in the experiments

In the merging stage, firstly the connections are listed according to their weight values in ascending order. Each voxel is initially assumed to be a segment. Starting the least weighted connection, the two segments at the ends of the connection are taken into consideration to merge them. In consideration, the boundary voxels between the two segments are determined at first. After that, the boundary voxels in the two segments are on-to-one paired mutually by the smallest weighted connection priority. If the all weight values of the connections between the voxels pairs are less than or equal the threshold parameter τ_{ang}, the segments are merged. If one of the two segments is a non-surface voxel, they are merged regardless of the merging criterion. A non-surface voxel can be merged once and is ignored in the future pairing processes.

2.3 The segmentation datasets

The two outdoor datasets Mosque and Train Station have been obtained from the study [20] and used in the tests of this work. The datasets have their segmentation ground-truth. The Mosque dataset includes 3,069,150 points, and its ground-truth has 42 segments. On the other hand, the Train Station dataset includes 2,742,237 points, and its ground-truth has 106 segments. In their source study, the voxel sizes of Mosque and Train Station are determined as 0.03 m and 0.045 m respectively. In the experiments of this work, the same values have been used as voxel size. In Figure 4, the original versions of the datasets and their colored ground truth are presented.

3. PROPOSED PLANE REFITTING METHOD

Determining the non-planar local points in the patches and refitting them are the main contributions of this paper. These processes are carried out before accomplishing the segmentation process in this paper, unlike some refinement methods in the literature. The plane refitting method proposed as an intermediate refinement process in this study consists of two stages. The first is determining the non-planar patches; and, the other is refitting the non-planar patches using the plane clustering process.

3.1 Determining the non-planar patches

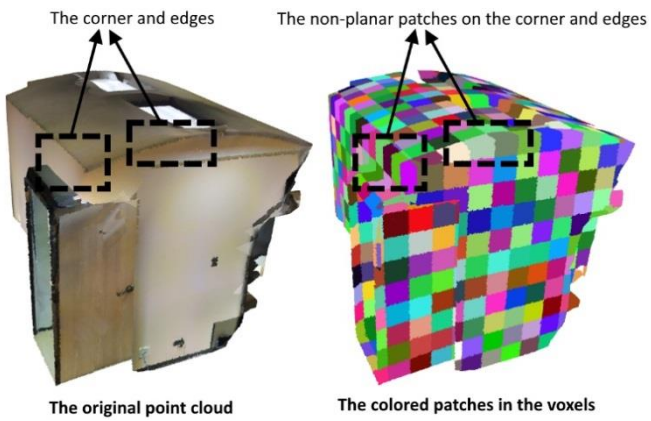

Figure 5. The examples of non-planar patches

In the literature, some point cloud segmentation algorithms [14, 32] split the point clouds by the octree-organization until the points in the octants reach the planar property. They use the standard deviation of the points from the fitting plane to decide that a point group is planar. In this work, this technique is used to determine whether a patch is planar or not. In the plane standard deviation and plane refitting stages, the non-surface patches should not be included in these operations. To compute the standard deviation σ_i of a fitting plane, first the dot productions of the spatial coordinates of the points p_l^i, which is lth point in the voxel V_i, and the barycenter X_i of them with n_i are calculated. In the way in Eq. (3) and (4), the normal axis values $\vec{p}_l^i$ and $\vec{X}^i$ of them are obtained, and then σ_i is calculated as in Eq. (5), where m_i is the number of points in V_i. If σ_i is bigger than the threshold parameter τ_σ, V_i is marked as non-planar as seen in Eq. (6). In Figure 5, some examples of non-planar patches can be seen on a point cloud sample obtained from the study [35].

$$\vec{X}^i = X_i \cdot n_i \tag{3}$$

$$\vec{p}_l^i = p_l^i \cdot n_i \tag{4}$$

$$\sigma_i = \sqrt{\frac{1}{k}\sum_{l}^{m_i}\left(\vec{p}_l^i - \vec{X}^i\right)^2} \tag{5}$$

$$\sigma_i > \tau_\sigma \Rightarrow V_i \text{ is non-planar} \tag{6}$$

It is difficult to set the threshold value τ_σ due to the point distribution change according to the data used. For this reason, the threshold selection is simplified in this work. To determine the threshold, the average standard deviation value $\bar{\sigma}$ calculated in Eq. (7) is used. Instead of determining the value τ_σ in an ambiguous range for users, determining c that refers to the Min-max [36] normalized τ_σ in the ranges [0, 1] scaled from the range $\bar{\sigma}$ to σ_{max} (the maximum possible standard deviation) where $\tau_\sigma > \bar{\sigma}$ and [-1, 0] scaled from 0 to $\bar{\sigma}$ where $\tau_\sigma < \bar{\sigma}$ helps users to determine a threshold value. The value σ_{max} is the half of the farthest distance in the voxel and calculated as in Eq. (8). The threshold parameter c refers to the normalized form of τ_σ as in Eqns. (9) and (10).

$$\bar{\sigma} = \frac{1}{m}\sum_{i}^{m}\sigma_i \tag{7}$$

$$\sigma_{max} = \frac{\sqrt{3}\cdot voxel\ size}{2} \tag{8}$$

$$c = \frac{\tau_\sigma - \bar{\sigma}}{\sigma_{max} - \bar{\sigma}}, \tau_\sigma > \bar{\sigma} \tag{9}$$

$$c = \frac{\tau_\sigma - \bar{\sigma}}{\bar{\sigma}}, \tau_\sigma < \bar{\sigma} \tag{10}$$

Instead of selecting τ_σ by the user, selecting c in the range [-1, 1] is pretty much easier. In our method, the parameter τ_σ is calculated as in Eq. (11). According to Eq. (11), τ_σ would be $\bar{\sigma}$ where $c = 0$. The case $c = 1$ means that all of the voxels are planar. On the other hand, $c = -1$ means that all of the voxels are non-planar. In Figure 6, the values $\bar{\sigma}$ and τ_σ (where $c = 0.05$) are illustrated on the standard deviation histograms of the datasets Mosque and Train Station.

$$\tau_\sigma = \begin{cases} \bar{\sigma}, & c = 0 \\ \bar{\sigma} + c \cdot (\sigma_{max} - \bar{\sigma}), & c > 0 \\ \bar{\sigma} \cdot (1 + c), & c < 0 \end{cases} \quad (11)$$

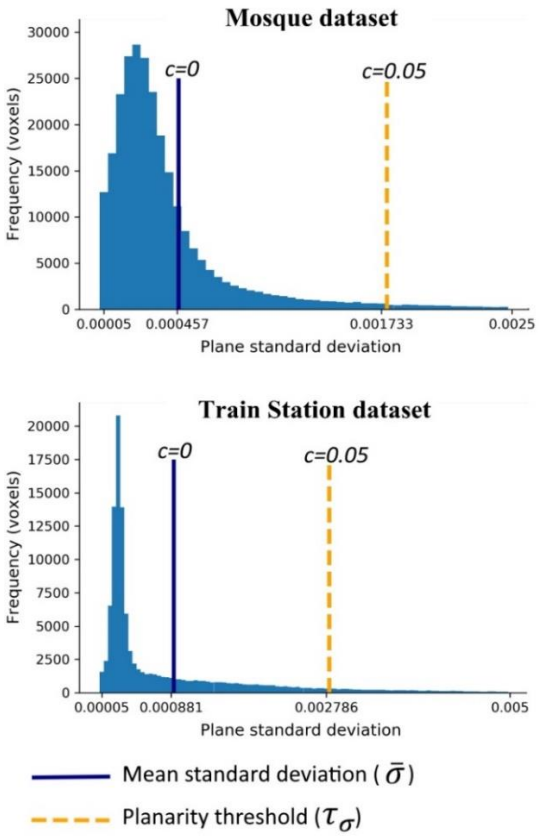

Figure 6. The histograms of standard deviations of patches on two datasets

After the threshold τ_σ is obtained using the parameter c, the non-planar local points in the patched are determined by operating the comparison in Eq. (6) for each patch.

3.2 Clustering-based plane refitting

In this stage, the patches marked as non-planar in the previous stage are clustered to planar sub-patches. The normal and barycenter properties of the voxels are replaced with those of the largest sub-patch. The clustering process is performed using the k-means algorithm [37]. K-means algorithm clusters the data given up to k clusters iteratively.

In traditional k-means clustering, appoint the data elements to the cluster whose cluster center is the nearest the element in each iteration, and the clusters and their centers are updated through the clustering until the termination criterion is satisfied [38]. In this work, the points are appointed to the clusters according to the proximity of the fitting planes of the clusters. In each iteration, the centers of normal axes (planes) of every cluster and the spatial coordinates of points on the normal axes are computed by the way in Eqns. (3) and (4). The points are appointed to the nearest plane and the clusters are redefined. The standard deviations of the new fitting planes of the clusters are computed. If all of the standard deviations of clusters are less than or equal to τ_σ or the number of points is less than 3 (non-surface), the refitting is terminated and the normal of the largest cluster is assumed as the normal of the voxel.

In the k-means clustering, the first operation is to determine the initial clusters. In this method, the initial clusters are determined by our specific method in this study. According to this method, the spatial coordinates of all points are transferred to the one-dimensional axis that stretches through the eigenvector corresponding to the highest eigenvalue. The new spatial values of the points through the axis are added to an empty list by sorting in ascending order. The list is divided into the k equal parts, and each part represents an initial cluster. In Algorithm 1, the k-means plane clustering is sketched.

The k cluster number parameter is selected 2 as initial because it is most probable that the patch falls within an edge. If the termination criterion cannot be met in the specified number of iteration (40 in our experiments), there will be no refitting. In this case, the number k is changed with 3 due to the probability that the patch may be in the corner, and the clustering process starts over as a second round. If the termination criteria cannot be reached in the second round, the refitting process ends without refitting. In Figure 7, the states of two example patches in several iterations are represented with gray level colors.

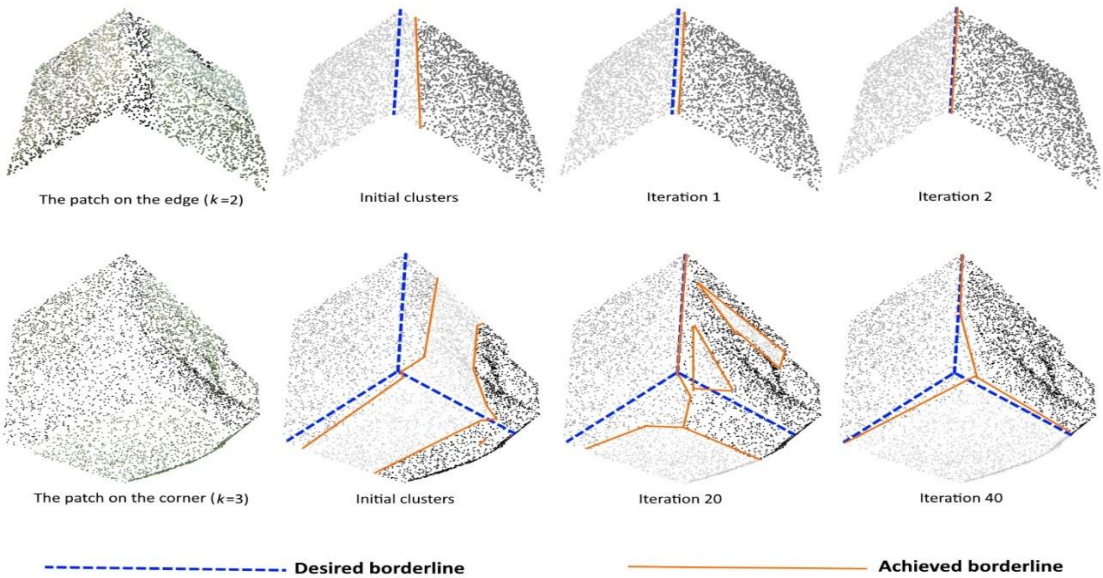

Figure 7. The k-means plane clustering examples

4. EXPERIMENTAL RESULTS

The experiments in this study were carried out using C++ programming language and performed on an Intel i9- 9900K CPU 3.6 GHz processor and 64GB of RAM. The segmentation successes of the two models which are the native BCVS and its extended version with the refitting process proposed in this study as an intermediate refinement method are compared in segmentation success and execution time.

Algorithm 1. The k-means plane clustering

Input: $P = \{\boldsymbol{p_1}, \boldsymbol{p_2}, \boldsymbol{p_3}, \cdots, \boldsymbol{p_m}\}$, k and $maxIter$

$\lambda_1 \geq \lambda_2 \geq \lambda_3 \leftarrow$ the eigenvalues of P
$\boldsymbol{e_1}, \boldsymbol{e_2}, \boldsymbol{e_3} \leftarrow$ the corresponding eigenvectors of P
$L \leftarrow \emptyset$
$m \leftarrow$ the size of P
for $l \leftarrow 1$ **to** m
    add $\boldsymbol{p_l} \cdot \boldsymbol{e_1}$ to L
sort L in ascending order
$Clusters = \{C_1, \cdots, C_k\} \leftarrow$ the parts of equally divided L by k

for $iter \leftarrow 1$ **to** $maxIter$
    for $t \leftarrow 1$ **to** k
        $\lambda_1 \geq \lambda_2 \geq \lambda_3 \leftarrow$ the eigenvalues of C_t
        $\boldsymbol{e_1}, \boldsymbol{e_2}, \boldsymbol{e_3} \leftarrow$ the corresponding eigenvectors of C_t
        $\boldsymbol{n_t} \leftarrow \boldsymbol{e_3}$
        $\boldsymbol{X_t} \leftarrow$ the barycenter of C_t
        $x_t \leftarrow \boldsymbol{X_t} \cdot \boldsymbol{n_t}$
    for $t \leftarrow 1$ **to** k
        $C_t \leftarrow \emptyset$
    for $l \leftarrow 1$ **to** m
        for $t \leftarrow 1$ **to** k
            $Dist_{l,t} \leftarrow |\boldsymbol{p_l} \cdot \boldsymbol{n_t} - x_t|$
        $nearest \leftarrow$ the smallest $Dist_l$ number
        $C_{nearest} \leftarrow C_{nearest} \cup \boldsymbol{p_l}$
    if $\forall \sigma_{plane}$ of $Clusters \leq \tau_\sigma$
        $max \leftarrow$ the largest cluster number
        if $max < 3$
            the patch is non-surface
            return
        $\boldsymbol{n_{patch}} \leftarrow \boldsymbol{n_{max}}$
        $\boldsymbol{X_{patch}} \leftarrow \boldsymbol{X_{max}}$
        return

4.1 The accuracy measurement as quantitative evaluation

As quantitative evaluations of the segmentation outputs, the accuracy measurement is used as described by Saglam et al. [20]. Quantitative measurement of a segmentation result is laborious compared to that for classification results because the number of segments and their labels of segmentation results are different from those of the ground-truth. For this reason, the result and ground-truth segments have been matched one-to-one mutually according to the method proposed in the study [39]. According to the matching method, the segments covered each other the most between the results, and ground-truth segments are matched in the first phase. In the second phase, the second mostly overlapped segments are looked for in mismatched segments. At the end of the second phase, some ground-truth segments may not be matched with any counter segment.

After the matching process, the number of overlapped points between the matched segments is considered as true segmented points (TP). The rate of TP to the number of all points in the ground-truth data gives the accuracy value.

4.2 Results

In the test stage, the angle parameter of the BCVS method is tested in the range 0-90° by increments of 1° for both with and without refitting. The c parameters in the refitting processes are set as 0.05. The refitting process is enforced to run the iteration of k-means at least two times in our experiments, although it meets the termination criterion in the first iteration.

On the graphics in Figure 8, the accuracy results of segmentations with the angle parameters (τ_{ang}) in the range 0-90° by increments of 1° for the BSCV algorithm without refitting and with refitting can be seen. In Table 1, the quantitative segmentation results of segmentations with the best parameters are shown as accuracy values. The colored labels of result segments are presented in Figure 9. In the figure, the main correct and incorrect regions are also enclosed by dashed frames and marked these regions as correct (√) or incorrect (×).

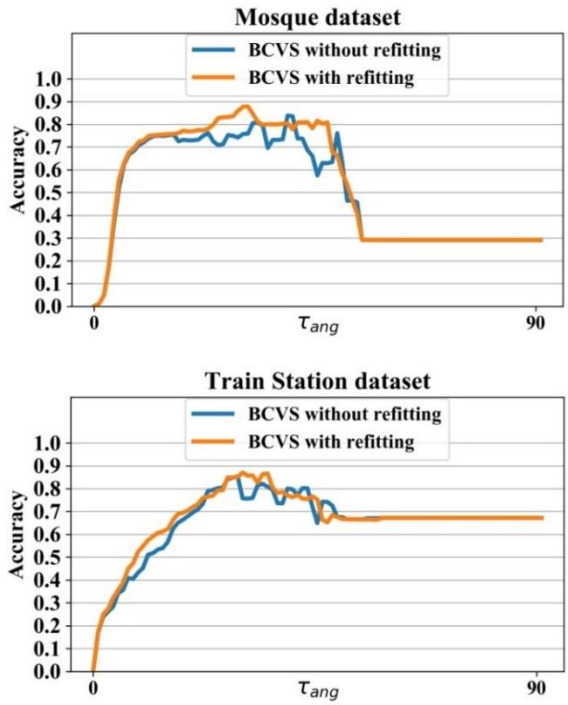

Figure 8. The accuracy lines of the methods with different angle parameters (τ_{max})

Table 1. The accuracy results of segmentation processes

Dataset/ Method	BCVS without refitting	BCVS with refitting
Mosque	0.8384	**0.8786**
Train Station	0.8536	**0.8707**

Table 2. The execution times (s) of the two models

Dataset/ Method	BCVS without refitting	BCVS with refitting
Mosque	2.429	2.550
Train Station	0.990	1.126

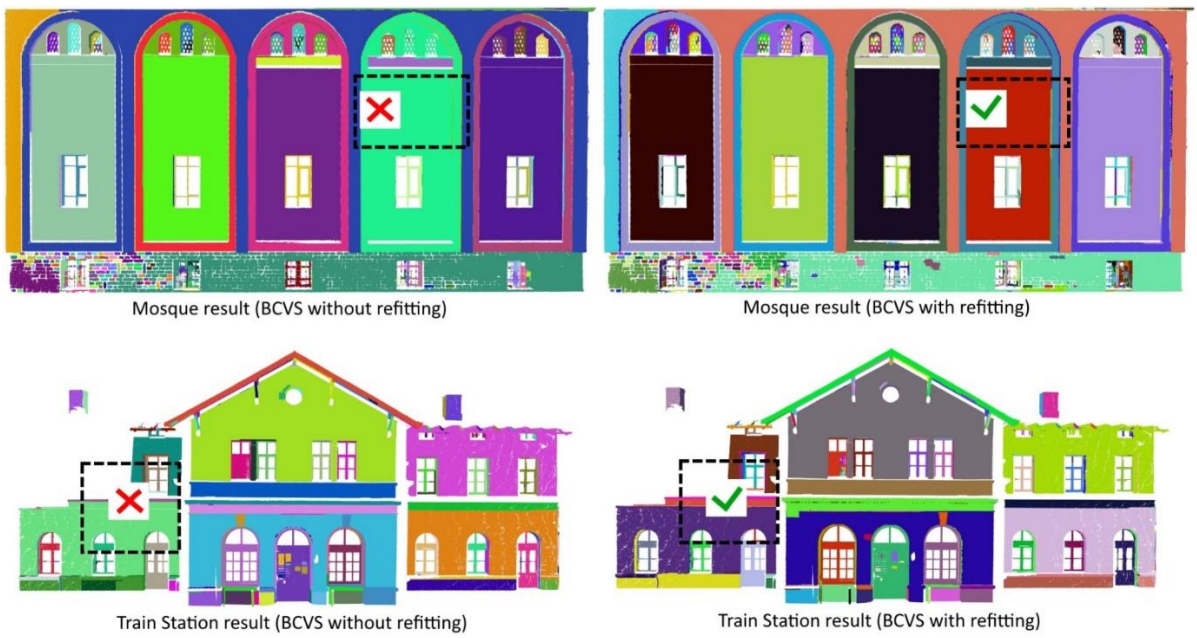

Figure 9. The colored labels of the result segments and the marks as correct (√) or incorrect (×)

In the experiments, the effect of the proposed intermediate process is also measured on the two datasets. In Table 2, the execution times of the two models are included as seconds (s). As seen in Table 2, the refitting process has little effect on execution time compared to that of the native model on the dataset used. Besides the structs of the points in the datasets, this effect also depends on the k-means parameter *k*. As the value of *k* increases, the non-planar patches (especially on the corners) are fit more precisely.

5. CONCLUSIONS

The voxelization as a sampling technique is a useful pre-processing for many point cloud processing applications. However, this technique has some deficiencies and the main one is that the variations in the feature such as local surface normal can be seen highly through the points (patch) in a voxel, against its usefulness such as data reduction, local neighbor finding, and outlier suppressing. The patches that have a high variation through points in it are named as non-planar patches in this work. The variation in the local surface normals in a patch may mislead a voxel-based segmentation method through the segmentation process. In this work, we determine non-planar patches using the plane standard deviations of patches and the mean of them. After non-planar patches are determined, the k-means clustering is applied to these patches in this study with some adaptations to cluster the patches into planar sub-patches. At the end of the clustering, if the process ends by satisfying the planarity criterion, the largest sub-patch replaces the spatial center and normal vector information of the patch with those of itself. The methods are applied to two point cloud segmentation datasets in a segmentation method. The results are examined quantitatively and visually. It is seen in the results that the refitting method increases the segmentation success by respectively approximately 4% and 2% on the Mosque and Train Station datasets. The only disadvantage of the proposed intermediate process on the inherent method is the increase in the execution time. On the other hand, the increase is not much compared to the duration of the inherent method. Besides the effect on the voxel-based segmentation process, the usage of the k-means clustering with the modifications for plane clustering in this paper can be useful for other point cloud processing applications in further studies.

ACKNOWLEDGMENT

This work is supported by the Scientific and Technological Research Council of Turkey (TUBITAK) (Grant number: 119E012). This study was carried out in the scope of the Doctoral Thesis of Ali SAGLAM.

REFERENCES

[1] Mahmoudabadi, H., Olsen, M.J., Todorovic, S. (2016). Efficient terrestrial laser scan segmentation exploiting data structure. ISPRS Journal of Photogrammetry and Remote Sensing, 119: 135-150. https://doi.org/10.1016/j.isprsjprs.2016.05.015

[2] Vosselman, G., Dijkman, S. (2001). 3D building model reconstruction from point clouds and ground plans. International Archives of Photogrammetry and Remote Sensing, 34(3/W4): 37-43.

[3] Lari, Z., Habib, A. (2014). An adaptive approach for the segmentation and extraction of planar and linear/cylindrical features from laser scanning data. ISPRS Journal of Photogrammetry and Remote Sensing, 93: 192-212. https://doi.org/10.1016/j.isprsjprs.2013.12.001

[4] Boyko, A., Funkhouser, T. (2011). Extracting roads from dense point clouds in large scale urban environment. ISPRS Journal of Photogrammetry and Remote Sensing, 66(6): S2-S12. https://doi.org/10.1016/j.isprsjprs.2011.09.009

[5] Clode, S., Kootsookos, P., Rottensteiner, F. (2004). The automatic extraction of roads from lidar data. ISPRS.

[6] Mayr, A., Rutzinger, M., Bremer, M., Oude Elberink, S., Stumpf, F., Geitner, C. (2017). Object-based classification of terrestrial laser scanning point clouds for

landslide monitoring. Photogrammetric Record, 32(160): 377-397. https://doi.org/10.1111/phor.12215

[7] Morsdorf, F., Meier, E., Allg, B., Daniel, N. (2003). Clustering in airborne laser scanning raw data for segmentation of single trees. ISPRS Working group III/3 Workshop 3-D Reconstruction from Airborne Laserscanner and InSAR Data. https://doi.org/10.2307/20708825

[8] Laefer, D.F., Pradhan, A.R. (2006). Evacuation route selection based on tree-based hazards using light detection and ranging and GIS. Journal of Transportation Engineering, 132(4): 312-320. https://doi.org/10.1061/(ASCE)0733-947X(2006)132:4(312)

[9] Lozes, F., Elmoataz, A., Lezoray, O. (2015). PDE-based graph signal processing for 3-d color point clouds: Opportunities for cultural heritage arts and found promising. IEEE Signal Processing Magazine, 32(4): 103-111. https://doi.org/10.1109/MSP.2015.2408631

[10] Pu, S., Vosselman, G. (2009). Knowledge based reconstruction of building models from terrestrial laser scanning data. ISPRS Journal of Photogrammetry and Remote Sensing, 64(6): 575-584. https://doi.org/10.1016/j.isprsjprs.2009.04.001

[11] Mitra, N.J., Nguyen, A., Guibas, L. (2004). Estimating surface normals in noisy point cloud data. International Journal of Computational Geometry & Applications, 14(4-5): 261-276. https://doi.org/10.1142/s0218195904001470

[12] Weinmann, M., Jutzi, B., Hinz, S., Mallet, C. (2015). Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers. ISPRS Journal of Photogrammetry and Remote Sensing, 105: 286-304. https://doi.org/10.1016/j.isprsjprs.2015.01.016

[13] Xu, Y., Yao, W., Tuttas, S., Hoegner, L., Stilla, U. (2018). Unsupervised segmentation of point clouds from buildings using hierarchical clustering based on gestalt principles. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 11(11): 4270-4286. https://doi.org/10.1109/JSTARS.2018.2817227

[14] Vo, A.V., Truong-Hong, L., Laefer, D.F., Bertolotto, M. (2015). Octree-based region growing for point cloud segmentation. ISPRS Journal of Photogrammetry and Remote Sensing, 104: 88-100. https://doi.org/10.1016/j.isprsjprs.2015.01.011

[15] Su, Y.T., Bethel, J., Hu, S. (2016). Octree-based segmentation for terrestrial LiDAR point cloud data in industrial applications. ISPRS Journal of Photogrammetry and Remote Sensing, 113: 59-74. https://doi.org/10.1016/j.isprsjprs.2016.01.001

[16] Poux, F., Billen, R. (2019). Voxel-based 3D Point Cloud Semantic Segmentation: Unsupervised geometric and relationship featuring vs deep learning methods. ISPRS International Journal of Geo-Information, 8(5): 213. https://doi.org/10.3390/ijgi8050213

[17] Wang, M., Tseng, Y.H. (2011). Incremental segmentation of lidar point clouds with an octree-structured voxel space. Photogrammetric Record, 26(133): 32-57. https://doi.org/10.1111/j.1477-9730.2011.00624.x

[18] Bassier, M., Bonduel, M., Van Genechten, B., Vergauwen, M. (2017). Segmentation of large unstructured point clouds using octree-based region growing and conditional random fields. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 42(2W8): 25-30. https://doi.org/10.5194/isprs-archives-XLII-2-W8-25-2017

[19] Xu, Y., Hoegner, L., Tuttas, S., Stilla, U. (2017). Voxel- and graph-based point cloud segmentation of 3D scenes using perceptual grouping laws. ISPAn, 41: 43-50. https://doi.org/10.5194/isprs-annals-IV-1-W1-43-2017

[20] Saglam, A., Makineci, H.B., Baykan, N.A., Baykan, Ö.K. (2020). Boundary constrained voxel segmentation for 3D point clouds using local geometric differences. Expert Systems with Applications, 157: 113439. https://doi.org/10.1016/j.eswa.2020.113439

[21] Papon, J., Abramov, A., Schoeler, M., Worgotter, F. (2013). Voxel cloud connectivity segmentation - Supervoxels for point clouds. 2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, 2027-2034. https://doi.org/10.1109/CVPR.2013.264

[22] Rusu, R.B., Blodow, N., Beetz, M. (2009). Fast point feature histograms (FPFH) for 3D registration. 2009 IEEE International Conference on Robotics and Automation, Kobe, pp. 3212-3217. https://doi.org/10.1109/ROBOT.2009.5152473

[23] Stein, S.C., Schoeler, M., Papon, J., Worgotter, F. (2014). Object partitioning using local convexity. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 304-311. https://doi.org/10.1109/CVPR.2014.46

[24] Zhu, Q., Li, Y., Hu, H., Wu, B. (2017). Robust point cloud classification based on multi-level semantic relationships for urban scenes. ISPRS Journal of Photogrammetry and Remote Sensing, 129: 86-102. https://doi.org/10.1016/j.isprsjprs.2017.04.022

[25] Verdoja, F., Thomas, D., Sugimoto, A. (2017). Fast 3D point cloud segmentation using supervoxels with geometry and color for 3D scene understanding. Proceedings - IEEE International Conference on Multimedia and Expo, pp. 1285-1290. https://doi.org/10.1109/ICME.2017.8019382

[26] Li, M., Sun, C. (2018). Refinement of LiDAR point clouds using a super voxel based approach. ISPRS Journal of Photogrammetry and Remote Sensing, 143: 213-221. https://doi.org/10.1016/j.isprsjprs.2018.03.010

[27] Jung, J., Che, E., Olsen, M.J., Shafer, K.C. (2020). Automated and efficient powerline extraction from laser scanning data using a voxel-based subsampling with hierarchical approach. ISPRS Journal of Photogrammetry and Remote Sensing, 163: 343-361. https://doi.org/10.1016/j.isprsjprs.2020.03.018

[28] Poux, F., Mattes, C., Kobbelt, L. (2020). Unsupervised segmentation of indoor 3D point cloud: Application to object-based classification. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 44(W1-2020): 111-118. https://doi.org/10.5194/isprs-archives-XLIV-4-W1-2020-111-2020

[29] Rabbani, T., van den Heuvel, F., Vosselman, G. (2006). Segmentation of point clouds using smoothness constraint. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences - Commission V Symposium “Image Engineering and Vision Metrology”, 36(5): 248-253. https://doi.org/10.1111/1750-3841.12802

[30] Shakarji, C.M. (1998). Least-squares fitting algorithms of the NIST algorithm testing system. Journal of Research of the National Institute of Standards and Technology, 103: 633-641. https://doi.org/10.6028/jres.103.043
[31] Xu, Y., Tuttas, S., Hoegner, L., Stilla, U. (2018). Voxel-based segmentation of 3D point clouds from construction sites using a probabilistic connectivity model. Pattern Recognition Letters, 102: 67-74. https://doi.org/10.1016/j.patrec.2017.12.016
[32] Wang, M., Tseng, Y.H. (2010). Automatic segmentation of lidar data into coplanar point clusters using an octree-based split-and-merge algorithm. Photogrammetric Engineering & Remote Sensing, 76(4): 407-720. https://doi.org/10.14358/PERS.76.4.407
[33] Canaz Sevgen, S. (2019). Airborne LiDAR data classification in complex urban area using random forest: a case study of Bergama, Turkey. International Journal of Engineering and Geosciences, 4(1): 45-51. https://doi.org/10.26833/ijeg.440828
[34] Gonzalez, R.C., Woods, R.E. (2007). Digital Image Processing (3rd Edition). Pearson International Edition.
[35] Armeni, I., Sener, O., Zamir, A.R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S. (2016). 3D semantic parsing of large-scale indoor spaces. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, pp. 1534-1543. https://doi.org/10.1109/CVPR.2016.170
[36] Jain, A., Nandakumar, K., Ross, A. (2005). Score normalization in multimodal biometric systems. Pattern Recognition, 38(12): 2270-2285. https://doi.org/10.1016/j.patcog.2005.01.012
[37] Jain, A.K., Dubes, R.C. (1988). Algorithms for Clustering Data. Prentice Hall.
[38] Saglam, A., Akhan Baykan, N. (2019). Evaluating the attributes of remote sensing image pixels for fast k-means clustering. Turkish Journal of Electrical Engineering and Computer Science, 27(6): 4188-4202. https://doi.org/10.3906/elk-1901-190
[39] Awrangjeb, M., Fraser, C.S. (2014). An automatic and threshold-free performance evaluation system for building extraction techniques from airborne LIDAR data. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 7(10): 4184-4198. https://doi.org/10.1109/JSTARS.2014.2318694

NOMENCLATURE

σ_i	Plane standard deviation.
τ_σ	Plane standard deviation threshold.
c	The refitting parameter. The value of this parameter can be in the range [-1 1]. This parameter indicates the normalized form of τ_σ.
τ_{max}	The angular segmentation parameter of the BCVS method. The value of this parameter can be in the range [0 90].