Received 27 March 2023, accepted 19 April 2023, date of publication 26 April 2023, date of current version 4 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3270709

RESEARCH ARTICLE

Indoor Point Cloud Segmentation Using a Modified Region Growing Algorithm and Accurate Normal Estimation

WEI WANG[1,2], YI ZHANG[2], GENGYU GE[1,2], QIN JIANG[1,2], YANG WANG[1,2], AND LIHE HU[1,2]
[1]College of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
[2]School of Advanced Manufacturing Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China

Corresponding author: Yi Zhang (zhangyi@cqupt.edu.cn)

This work was supported in part by the National Science Fund for Distinguished Young Scholars of China under Grant 61803058 and Grant 61703067, and in part by the Research Project of Chinese Disabled Persons' Federation-on Assistive Technology under Grant 2022CDPFAT-01.

ABSTRACT With the development of 3D sensors, 3D point cloud data can now be obtained conveniently. Therefore, it is crucial to process point cloud data automatically. Region growing is a commonly used algorithm to segment point clouds, which greatly depends on the accuracy of points' normals and requires tuning two thresholds; i.e., the increment of curvature (σ_{th}) and normal angles (θ_{th}). In this paper, we improve the region growing algorithm in two ways: Accurate normal estimation and strengthening the region growing criteria. For the first aspect, principal component analysis (PCA) is utilized to estimate the initial normals of the point cloud. Then, the points are divided into regular points (RP) and sharp feature points (SFP), according to their initial normals. A robust estimator-based PCA is then applied to refine the SFP normals. For the latter aspect, non-connected and non-coplanar points are detected and ignored when region grows. Finally, the segmentation performance of the proposed method is evaluated using internal and external indices. The results indicate that the proposed method can accurately estimate the point normals within an acceptable time, and obtain a better result than the classic PCA-based region growing algorithm and advanced DetMM-based methods.

INDEX TERMS Normal estimation, region growing, PCA, robust estimator, point cloud segmentation.

I. INTRODUCTION

The amount of point cloud data has been growing rapidly as 3D sensors (such as stereo cameras, laser scanning, and so on) have become more advanced. These data can be used for robot navigation (unmanned vehicles) [1], 3D geometry modeling [2], and default detection, among other things. While low-precision sensors can only collect tens of thousands of points, high-precision ones can collect millions. It takes a lot of computation to manage this enormous amount of data. In general, 2D LiDAR and RGB-D cameras may provide indoor mobile robots with 2D and 3D environmental information, respectively. Instead of 2D sensors, 3D sensors are required for the indoor robot to comprehend its environment at the semantic level. Robots can easily acquire 3D models or maps of indoor environments using 3D LiDAR-based techniques like LOAM [3], RTAB-Map [4], and HDL_GRAPH_SLAM [5], as well as RGB-D camera-based techniques like KinectFusion [6] and ElasticFusion [7].

This paper mainly aims at point cloud post-processing for 3D models of indoor environments. The primary artificial geometries that robots see indoors include planes with regular shapes and curved surfaces. So, plane segmentation plays a vital role in indoor point cloud processing, which has been widely used in reverse engineering [8], object recognition [9], augmented reality [10], heritage preservation [11], and so on. Therefore, it has become a research hotspot to extract plane primitives from unorganized point clouds. The methods of plane detection can be divided into four categories, i.e., (1) edge-based segmentation, (2) region growing segmentation, (3) model-based segmentation, (4) and unsupervised clustering methods [12]. Analyses of these methods are provided in the following.

The associate editor coordinating the review of this manuscript and approving it for publication was Zahid Akhtar.

Edge-based segmentation technology [13] consists of two steps: edge detection and point clustering. Although these methods can rappidly segment point clouds, they suffer from low accuracy due to the existence of noise and the uneven density of point clouds. Model-fitting based methods are mainly used to segment particular man-made objects by fitting geometric primitives such as planes, cylinders, and spheres. However, it is difficult for these methods to segment complex shapes or realize fully automated implementations, as details can not always be modeled by easily recognizable geometrical shapes. Additionally, RANSAC may fit a plane that does not actually exist.

Meanwhile, numerous unsupervised machine learning (ML) algorithms have been designed to segment point clouds, such as mean-shift [14], dbscan [15], k-means, and hierarchical clustering [16], homogenization clustering network [17], graph clustering [18], etc. However, it is crucial to set parameters and define features for these machine-learning techniques. Region growing segmentation [19], [20] has been designed to segment 2D data (images). To segment point clouds, one popular technique is transforming 3D points into a 2D domain [21] and using image processing algorithms to segment images, then project the result onto the point clouds. Later, it was developed to directly process 3D point clouds. Starting from one or more seed points, it grows around the neighboring points, according to pre-defined criteria, which is commonly used to extract plane primitives.

Region growing algorithms for point clouds typically consist of two steps: the selection of the seed and the growing. The seed can be selected according to the distance between points and the fitting plane [22] or the curvature of the points [23], [24], and the region grows according to the angle increment of point normals [22]. Moreover, the distance between the k-NN (k-Nearest Neighbors) and the fitting plane [24] has been introduced as a criterion of region growing.

The key contributions of this paper are as follows: (1) More accurate normals are estimated within a reasonable time-consumption; (2) Non-coplanar and non-connective points are detected, the former will be removed from the k-NN and the latter will not be added to the seed set as the region expands; (3) The threshold σ_{th} is set to the 95th percentile of the curvature of the k-NN, such that the desired segmentation can be easily obtained by tuning θ_{th}. The remainder of this paper is organized as follows: Section II describes the proposed method, which consists of normal estimation and region growing. Section III introduces the evaluation metrics. Section IV provides the experimental results. Section V makes conclusions including limitations of the proposed method.

II. METHODOLOGY

As mentioned above, the region growing algorithm involves the selection of seeds and the criteria of region growing. It is a frequently used algorithm that is included in the point cloud library (PCL) [26] to select the seeds by curvature increments and the region grows according to the angles between the seeds and their neighbors' normals [25]. However, the segmentation result greatly depends on the collaborative adjustment of two thresholds (σ_{th} and θ_{th}), and there is no set rule for adjusting these thresholds. So, it is difficult to set a pair of appropriate thresholds to achieve better segmentation. Furthermore, the normals and curvatures play crucial roles, which are commonly estimated by conducting PCA (Principal Component Analysis) on the k-NN of a point. To estimate the normal for n points, k-NN searching needs to be performed n times, which is time-consuming. Kd-tree is often used to accelerate k-NN searching, which has a time complexity of $\mathcal{O}(\log n)$ [27].

The proposed method consists of two parts: Accurate normal estimation and modified region growth, as shown in Figure 1.

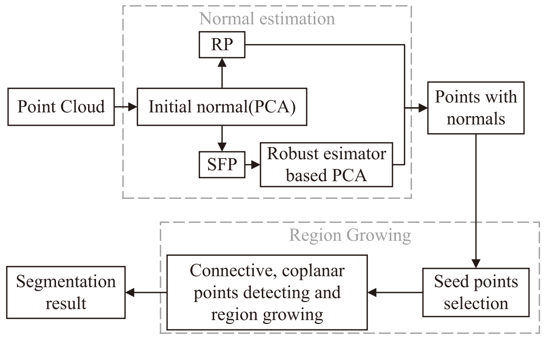

FIGURE 1. Framework of the proposed method.

A. NORMAL AND CURVATURE ESTIMATION

Normals (including curvatures) are basic features of a point cloud that are widely used in many applications. The region growing algorithm selects a seed according to the curvature, and the region expanded according to the angle difference between the normals of the reference point and its neighborhoods. If the normals and curvatures are not accurate, the segmentation will be rough and, consequently, may even be false.

1) PRINCIPAL COMPONENT ANALYSIS

PCA [28] is commonly used to estimate point normals. Let a point cloud be denoted as a set $\{p_i\}_{i=1,2,\cdots,n}, p_i = (x_i, y_i, z_i)$ is a point in the point cloud. Then the k-NN of p_i can be denoted as $\{p_j\}_{j=1,2,\cdots,k}$, where n is the number of points and k is the number of pre-defined neighbors. The covariance matrix Cov can be calculated by Equation (1):

$$Cov_{3\times3} = \frac{1}{k}\sum_{j=1}^{k}(p_j - \bar{p})^T(p_j - \bar{p}), \quad \bar{p} = \frac{1}{k}\sum_{j=1}^{k} p_j, \tag{1}$$

where $\bar{p}$ is the centroid (arithmetic mean) of the k-NN. By performing SVD (singular value decomposition) on Cov, the eigenvalues λ ($\lambda_0 \leq \lambda_1 \leq \lambda_2$) and the corresponding eigenvectors v (v_0, v_1, and v_2) can be obtained. Then, the

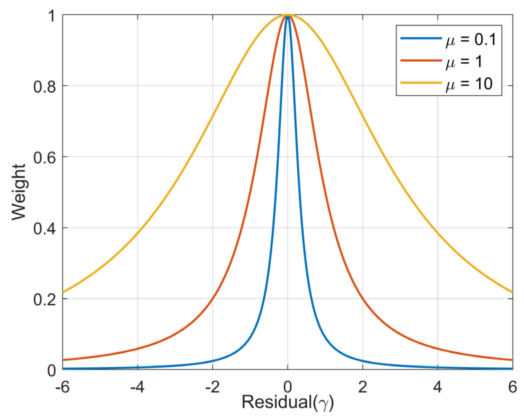

FIGURE 2. **The weighting function of Geman-McClure estimator.**

eigenvector v_0 is regarded as the approximate normal of p_i. The span of the eigenvectors v_1 and v_2, i.e. $span(v_1, v_2)$, is the fitted plane. The curvature is defined as the ratio of the minimum eigenvalue to the sum of three eigenvalues [29], as shown in Equation (2):

$$\sigma(p_i) = {}^{\lambda_0}/_{(\lambda_0+\lambda_1+\lambda_2)}, \quad \lambda_0 \leq \lambda_1 \leq \lambda_2. \tag{2}$$

Due to the influence of noise or non-coplanar points, the estimated normals are wildly inaccurate, especially when the points are close to the edge or corner areas. Many optimization methods have been proposed to obtain more accurate normals.

2) ITERATION-WEIGHTED PCA

Weighted PCA is an alternative PCA method, the key point of which is the calculation of weights, as shown in Equation (3). Iteration-weighted PCA [30] utilizes M-estimator to obtain weights.

$$\vec{n}_i = \underset{\|\vec{n}_i\|=1}{\arg\min} \sum_{j=1}^{k} m_j \left(\vec{n} \cdot (p_j - p_i)\right), \tag{3}$$

M-estimator is a general class of estimators where we can hope to find estimators with good robustness properties by choosing an appropriate function ρ. It is defined implicitly as the solution of the equation [31]

$$\sum_{i=1}^{n} \rho \left(\frac{x_i - \hat{\theta}}{\hat{\sigma}} \right) = 0. \tag{4}$$

Some commonly used M-estimators and their corresponding weighting functions are summarized in [32]. A scaled version of the Geman-McClure estimator parameterized by a scalar μ, as shown in Figure 2, is applied to calculate the weights as shown in Equation (5):

$$m_j = \left({}^{\mu}/_{(\mu+\gamma_j)}\right)^2, \tag{5}$$

where $\gamma_j = \overrightarrow{p_i p_j} \cdot \vec{n}$ and μ is the scale factor.

Figure 2 shows that the weight function decays more quickly the smaller the value of μ. So, iteration-weighted PCA decreases the value of μ by $\mu = \mu/1.01$ to refine the point normals. Despite not being able to estimate point curvature, it can produce precise normals. It is solely used in this article to compare the accuracy of normal estimation since it can not be directly used for region growing algorithm.

3) ROBUST ESTIMATOR-BASED PCA

Robust estimators, such as MVE/MCD [33], FastMCD [34], DetMCD [36], and DetMM [37], can be used to estimate the location (μ) and scatter (Σ) of data. Then, outliers can be detected by comparing the Mahalanobis Distance (MD) to the 97.50th percentile of the chi-square (χ^2) distribution with three degrees of freedom (i.e., $\chi^2_{3,0.975} = 3.075$), as shown in Equation (6):

$$p_j \text{ is } \begin{cases} inlier \;\; if \;\; MD_j^2 \leq \chi^2_{3,0.975} \\ outlier \qquad otherwise \end{cases}, \tag{6}$$

where $MD = \sqrt{(X-\mu)\,\Sigma^{-1}\,(X-\mu)^T}$ and $X = \{p_1, p_2, \cdots, p_k\}^T$. After detecting outliers, PCA is only performed on inliers to estimate the normal of the p_i.

Khaloo et al. have used DetMM to estimate point normals [38]; however, DetMM and DetMCD do not always work well, as a result of the so-called process of scaling the variables, as shown in Equation (7):

$$Z_j = {}^{(X_j - \text{med}(X_j))}/_{Q_n(X_j)}, \tag{7}$$

where X_j is the column vector of X, med(X_j) is the median value of X_j, and $Q_n(X_j)$ is the scale estimator of X_j by $Q_n(x) = 2.2219\{|x_i - x_j; i < j|\}_{(\kappa)}$, where $\kappa = \binom{h}{2}$ and $h = \lfloor \frac{n}{2} \rfloor + 1$ [39], which means the first quartile of inter-point distances. Obviously, it can not be guaranteed that the denominator, $Q_n(X_j)$, is not equal to zero. In addition, estimating the normal of each point requires running DetMM n (number of points) times, which takes a lot of time.

In summary, PCA can quickly but inaccurately obtain points' normals; Iteration-weighted PCA may obtain accurate normals but does not estimate curvatures; DetMM-based PCA can reliably estimate point normals but has a high time cost and occasionally fails. So, this paper comprehensively weighs both accuracy and time consumption and proposed a new normal estimation approach.

4) PROPOSED NORMAL ESTIMATION METHOD

This paper aims to segment indoor point clouds using a region growing algorithm. Although the accuracy of the point's normal has a significant impact on the segmentation, it is not worthwhile to take dozens of minutes to obtain better points' normals. By comprehensively weighing both accuracy and time consumption, a normal estimation method has been proposed to obtain normals more precisely compared to PCA and faster compared to DetMM-based PCA. First, PCA is used to get the initial points' normals. Then, the points are divided into SFP (sharp feature points) and RP (regular points). Finally, robust estimator-based PCA is used to estimate the normals of SFP.

Let the initial normals be denoted as $\vec{n}_{init} = (\vec{n}_1, \vec{n}_2, \cdots, \vec{n}_n)^T$. Then the angle, θ_{ij}, between the normals $\vec{n}_i$ and $\vec{n}_j$ can be calculated by Equation (8):

$$\theta_{ij} = \left\| \arccos(\vec{n}_i \cdot \vec{n}_j) \right\|, \tag{8}$$

where $\vec{n}$ is a unit normal.

Assume a point is p_i, its normal is $\vec{n}_i$, and its k-NN are $p_1, p_2, \cdots, p_k$, the normals of which are $\vec{n}_1, \vec{n}_2, \cdots, \vec{n}_k$. The angles between the normal of p_i and its k-NN are denoted as $\theta_i = (\theta_{i1}, \theta_{i2}, \cdots, \theta_{ik})$; then, whether a point is an SFP or RP is determined by Equation (9):

$$p_i \text{ is} \begin{cases} \text{RP} & \textit{if } Q_n(\theta_i) \le \theta_{th} \\ \text{SFP} & \textit{otherwise} \end{cases}, \tag{9}$$

where θ_{th} is consistent with the threshold θ_{th} of the region growing algorithm (i.e., $\theta_{th} = 5°$).

Based on this criterion, point clouds sampled from CAD-like mesh models were utilized to test and visualize the RP and SFP, as shown in Figure 3.

Actually, the normals of RP are relatively accurate, only those of SFP need to be refined. Thus, robust estimator-based PCA is used to refine the SFP normals. Although Khaloo et al. has adopted DetMM-based PCA to estimate point normals [38], they did not consider the applicability of DetMM. As discussed above, DetMM is not suitable for data whose scaling variables are equal to zero. To overcome this disadvantage of DetMM, DetMCD is used to refine the normals of SFP.

MCD (minimum covariance determinant) estimator is a very robust estimator of multivariate location and scatter with the maximum breakdown value of 50%. As previously defined, $p_1, p_2, \cdots, p_k$ are k-NN of p_i. The goal of MCD is to choose h points from k-NN so that the determinant of its covariance matrix is minimized, where $\lfloor (k+p+1)/2 \rfloor \le h \le k$. However, the computation is rather large due to the vast number of possible combinations $\binom{h}{k}$. FastMCD significantly accelerated the computation of MCD and made it widely applied in various fields. DetMCD is a deterministic version of MCD and runs even faster than FastMCD. So, the DetMCD estimator is used to find inliers, and PCA is applied to the inliers to refine the normals of SFP. The algorithm for normal estimation is provided in Algorithm 1.

This paper compares the proposed method to other commonly used normal estimation algorithms in terms of MSAE [40] and time consumption, namely PCA, 2-jets [35], iteration-weighted PCA, and DetMM-based PCA. Figure 4 shows the MSAE on cuboid and dodecahedron meshes (the other models are similar), where noise level implies the standard deviation proportional to the diagonal length of the bounding box. The average time consumption on point clouds sampled from cube meshes with varied noise levels is recorded in table 1. PCA and 2-jets were evaluated with the Computational Geometry Algorithms Library (CGAL), and iteration-weighted PCA was evaluated using MPI-accelerated online code. As a result, they were mostly utilized in accuracy comparison (i.e., MSAE). All of the methods were run on a laptop with an AMD R7-4800U 4.2 GHz processor. The parameter k was set to 50.

Algorithm 1 Normal Estimation

```
Input: Point cloud: $\{P\}$;
    the number of neighbors: $k$;
    Threshold: $\theta_{th}$;
Build a $k-d$ tree: tree = KDTreeSearch($P$);
Find the $k$-NN of each point:
    knn_idx = knnsearch(tree, $P$, $k$);
Calculating the initial normals:
    [$\vec{n}$, $\sigma$] = NormEstim_PCA($P$, knn_idx);
Calculating the normal angle difference between $p_i$ and
its $k$-NN:;
    AnglDiffer = CompAnglDiffer($\vec{n}$, knn_idx);
Find SFP:
    $\theta \leftarrow$ 25th percentile of AnglDiffer;
    idx_sharp $\leftarrow$ find($\theta > \theta_{th}$);
for $i$ = 1 to size(idx_sharp) do
    idx $\leftarrow$ idx_sharp$_i$;
    knn $\leftarrow$ knn_idx$_{\text{idx}}$;
    $p_{\text{knn}} \leftarrow P_{\text{knn}}$;
    [$\vec{n}_{\text{idx}}$, $\sigma_{\text{idx}}$] = NormEstim_DetMCD_PCA($p_{\text{knn}}$);
Output: $\vec{n}$, $\sigma$.
```

TABLE 1. Time consumption (in seconds) of various methods.

Noise Level	[28]	[35]	[30]	[38]	Proposed
0.25%	5.253	9.235	6.858	602.491	**38.58**
1.20%	6.858	10.259	8.075	621.695	**37.613**
2.40%	7.567	8.236	7.567	710.536	**43.817**

[28] represents PCA;
[35] represents 2-jets;
[30] represents iteration-weighted PCA;
[38] represents DetMM-based PCA.

In Figure 4, it can be seen that the proposed method obtained the second-lowest MSAE in terms of accuracy. And in Table 1, the proposed method was an order of magnitude faster than DetMM [38], and could generally estimate point normals within one minute in terms of time consumption. PCA [28] and 2-jets [35] are fast but inaccurate. DetMM [38] is robust to noise, but it takes the most time. Iteration-weighted PCA [30] was fast and accurate, however, it dose not estimate curvature, as previously stated.

B. REGION GROWING ALGORITHM

As proposed in the literature [25], the classical region growing algorithm involves the expansion of two point sets: Region R and Seed S. First, the point with the minimum curvature value among the unexpanded points is selected as a seed and added to S. Then, the set S will expand according to the curvature constraint (i.e., $\Delta\sigma \le \sigma_{th}$). Meanwhile, the region R will grow around the k-NN of the seed according to the angle constraint (i.e., $\Delta\theta \le \theta_{th}$). Obviously, it is

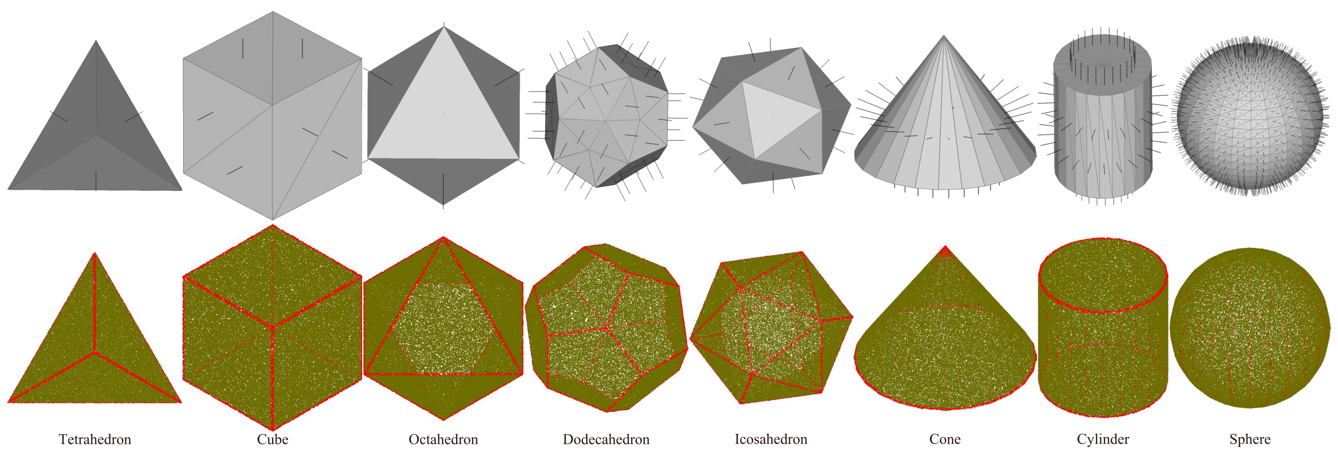

FIGURE 3. First row: CAD-like mesh models and the normal ground truth. Second row: Points sampled from mesh (SFP, red; RP, dark green).

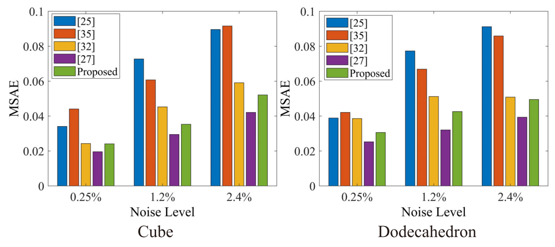

FIGURE 4. MSAE (normal errors) of points sampled from cuboid and dodecahedron meshes; noise level denotes the factor of the diagonal distance of the bounding box. [25] represents PCA; [35] represents 2-jets; [32] represents DetMM-based PCA; [27] represents iteration-weighted PCA.

necessary to set θ_{th} and σ_{th}, but there is no rule indicating how to set these values appropriately. Thus, it is both theoretically and empirically difficult to find a perfect pair of parameters.

It is preferable to set fewer parameters manually while upgrading the algorithm. As has been mentioned in the literature [41], σ_{th} should to be set as the 98th percentile of σ, while θ_{th} should be set to 15°. This setting, however, is not universally applicable. The values of σ_{th} and θ_{th} vary with different applications. Anyway, there is something to be learned from this strategy, which is to fix one parameter before adjusting the other. In any case, it is possible to learn from this strategy, which involves setting one parameter and then modifying the other.

In combination with this method, two sets of test experiments were designed on the dataset *region_growing_tutorial*, in which σ_{th} was set to the 95th and 98th percentiles of the curvatures of k-NN, respectively, and θ_{th} was tuned in the range of [2°, 10°]. When the curvature is set to the 95th or 98th percentile, the segmentation is nearly identical; however, $\theta_{th} = 2°$ leads to under-segmentation and $\theta_{th} = 8°$ leads to over-segmentation. As a result, σ_{th} is set to the 95th percentile of the curvatures of k-NN, then the desired segmentation can be achieved by tuning θ_{th} around 5°.

The classical region growth algorithm takes all k-NNs as candidate points when the sets R and S expand, regardless of whether they are connective or coplanar. Actually, if a neighboring point is far from p_i (i.e., non-connective), it should not be added to the seed S, and if a neighboring point is far away from the plane fitted by the p_i's k-NN (i.e., non-coplanar), it should not be added into the region R. Thus, criteria are proposed to judge whether neighbor points are coplanar or connective, as follows:

(1) **Connective points**: Not all k-NN are considered connected; only points that share 75% of their k-NN are considered connected. Let knn_i and knn_j be the indices of the k-NN of p_i and p_j, respectively. Then, the connectivity of p_i and p_j is defined by Equation (10):

$$p_j \text{ is} \begin{cases} \text{connective} & \textit{if } \text{knn}_i \cap \text{knn}_j \geq \text{knn}_{th} \\ \text{non-connective} & \textit{otherwise} \end{cases}, \quad (10)$$

where $\text{knn}_{th} = 75\% \cdot k$. Connectivity constraints can be changed by tuning the number of shared neighbors.

(2) **Coplanar points**: Upon the assumption that the noise obeys a Gaussian distribution, it is easy to understand that most k-NN are located nearby the plane fitted by them. As above, the k-NN of p_i is $\{p_j\}_{j=1,2,\cdots,k}$. Then, the residuals of the p_i's k-NN to the fitted plane are $\gamma_i = \{\gamma_{ij}\}_{j=1,2,\cdots,k}$, where $\gamma_{ij} = \|\overrightarrow{p_i p_j} \cdot \vec{n}_i\|$. A robust metric γ_{th} is introduced to detect non-coplanar points, as defined in Equation (11):

$$p_j \text{ is} \begin{cases} \text{coplanar} & \textit{if } \gamma_{ij} \leq \gamma_{th} \\ \text{non-coplanar} & \textit{otherwise} \end{cases}, \quad (11)$$

where $\gamma_{th} = \textit{median}\,(\gamma_i) + 2 \cdot Q_n\,(\gamma_i)$, and $Q_n\,(\gamma_i)$ is the same as in Equation (7).

Figure 5 illustrates the detected connective and coplanar points. The non-connective and non-planar points will not be added to corresponding sets S and R. As a result, the criteria for region growing are enhanced in this way.

Overall, the improved region growing algorithm was implemented as follows: After getting the initial normals

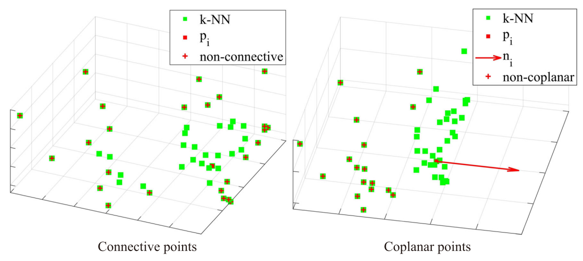

FIGURE 5. Detection of coplanar and connective points.

by PCA, points are classified as SFP or RP, and a robust estimator based PCA is used to refine the SFP's normals and curvatures. The σ_{th} is fixed as 95^{th} percentile of curvatures. Then, by detecting connective and coplanar points, the criteria of region growing are enhanced. The pseudo-code of the improved algorithm as shown in Algorithm 2.

Algorithm 2 Improved Region Growing Algorithm

```
Input: Point cloud: $\{P\}$;
    Point normals: $\{N\}$;
    Index of neighbor: knn_idx;
    Angle threshold: $\theta_{th}$;
Initialization: Region list: $R \leftarrow \emptyset$;
    Available points list: $\{A\} \leftarrow \{1, \cdots, \|P\|\}$;
while $\{A\}$ is not empty do
    Current region $\{R_c\} \leftarrow \emptyset$;
    Current seeds $\{S_c\} \leftarrow \emptyset$;
    Set $\sigma_{th}$ as the 95th percentile of point curvatures:
    $\sigma_{th} \leftarrow \sigma_{95th}$;
    The point with minimum curvature in $\{A\} \rightarrow P_{min}$;
    $\{S_c\} \leftarrow \{S_c\} \cup P_{min}$;
    $\{R_c\} \leftarrow \{R_c\} \cup P_{min}$;
    $\{A\} \leftarrow \{A\} \setminus P_{min}$;
    for $i$=1 to size $\{S_c\}$ do
        Find the $k$-NN of the current seed point: $\{B_c\} \leftarrow$
        knn_idx($S_c(i)$);
        for $j$=1 to size $\{B_c\}$ do
            knn_share =
            intersect(knn_idx($B_c(j)$),knn_idx($S_c(i)$));
            if length(knn_share)$< 75\% \cdot k$ then
                $\{B_c\} \leftarrow \{B_c\} \setminus B_c(j)$;
        $\Delta\theta_i$=ComputAngDiff($\vec{n}(S_c(i))$, $\vec{n}(B_c)$);
        $\gamma_i$=ComputResidual($\vec{n}(S_c(i))$, $\{B_c\}$);
        $\gamma_{th} = median(\gamma_i) + 2 \cdot Q_n(\gamma_i)$;
        for $j$=1 to size $\{B_c\}$ do
            The current neighbor point $P_j \leftarrow B_c\{j\}$;
            if $\{A\}$ contains $P_j$ and $\gamma_{ij} \leq \gamma_{th}$ and
            $\Delta\theta_{ij} < \theta_{th}$ then
                $\{R_c\} \leftarrow \{R_c\} \cup P_j$;
                $\{A\} \leftarrow \{A\} \setminus P_j$;
                if $\Delta\sigma_{ij} < \sigma_{th}$ then
                    $\{S_c\} \leftarrow \{S_c\} \cup P_j$;
    Add the current region to the global segment list:
    $\{R\} \leftarrow \{R\} \cup \{R_c\}$;
Sort $\{R\}$ in decsending order;
Output: $\{R\}$.
```

III. PERFORMANCE METRICS

To validate the segmentation performance, internal and external indices were utilized. *compactness* (CP), *separation* (SP), the *Davies–Bouldin index* (DBI), and the *Dunn validity index* (DVI) were among the internal indices, while *precision*, *recall*, and *F1-score* were among the exterior indices.

A. INTERNAL VALIDATION

Identical notations are used in the following equations, where k denotes the number of clusters, Ω_i represents the point set of the i^{th} cluster, $\|\Omega_i\|$ is the number of points in the i^{th} cluster, x_i represents a point, and w_i denotes the centroid of the i^{th} cluster. It is worth noting that the k used in this section is different from the number of neighbors k used above.

- **CP** measures the average distance between data in the same cluster, and a lower value is preferred. The calculation of CP is shown in Equation (12):

$$CP = \frac{1}{k}\sum_{i=1}^{k}\overline{CP}_i, \quad \overline{CP}_i = \frac{1}{\|\Omega_i\|}\sum_{x_i\in\Omega_i}\|x_i - w_i\|_2 . \tag{12}$$

- **SP** measures the degree of separation between clusters, and a larger value is preferred. The calculation of SP is shown in Equation (13):

$$SP = \frac{2}{k^2 - k}\sum_{i=1}^{k}\sum_{j=i+1}^{k}\left\|w_i - w_j\right\|_2 . \tag{13}$$

- **DBI** evaluates the clusters according to the ratio of within-cluster scatters to between-cluster separations. Good clusters are compact and separative, which means a lower value of DBI. The calculation of DBI is shown in Equation (14):

$$DBI = \frac{1}{k}\sum_{i=1}^{k}\max_{j\neq i}\left(\frac{\overline{\Omega_i} + \overline{\Omega_j}}{\left\|w_i - w_j\right\|_2}\right),$$
$$\overline{\Omega_i} = \frac{2}{\|\Omega_i\|(\|\Omega_i\| - 1)}\sum_{i=1}^{\|\Omega_i\|}\sum_{j=i+1}^{\|\Omega_i\|}\left\|x_i - x_j\right\|_2 . \tag{14}$$

- **DVI** quantifies both the degree of compactness and separation of clusters. Contrary to DBI, a larger value of DVI indicates that the cluster is more compact and separated.

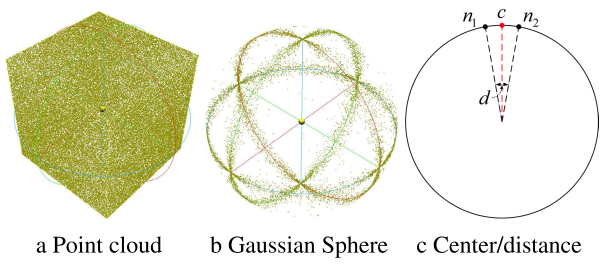

FIGURE 6. Projecting a point cloud onto a Gaussian sphere and the definition of center and distance.

The calculation of DVI is shown in Equation (15):

$$DVI = \frac{\min\limits_{0<m\neq n<k}\left\{\min\limits_{\substack{x_i\in\Omega_m\\ x_j\in\Omega_n}}\left(\left\|x_i-x_j\right\|\right)\right\}}{\max\limits_{0<m<k}\left\{\max\limits_{x_i,x_j\in\Omega_m}\left(\left\|x_i-x_j\right\|\right)\right\}} \tag{15}$$

To effectively evaluate clusters using the aforementioned internal indices, the point clouds were projected onto a Gaussian sphere [42], which clusters points with similar normals. Let the point cloud be $\{P\} = \{p_i\}_{i=1,2,\cdots,n}$ and normals be $\{n_i\}_{i=1,2,\cdots,n}$; then, the projection point of $\{P\}$ on the Gaussian sphere can be defined as $G(P) = \{n_i\}_{i=1,2,\cdots,n}$. Points with the same or similar normals will be projected to the same or near neighboring points on the Gaussian sphere, whereas points with significant variances in normals will be projected farther away. Meanwhile, the centroid and distance calculations should be suitable for the Gaussian sphere. Figure 6 illustrate the point projection and the definition of center and distance: Figure 6a shows the point cloud sampled from a box model, Figure 6b is the projection of point cloud on the Gaussian Sphere, Figure 6c illustrates the definition of center of two points and the distance between them.

The center points and distance can be calculated by Equations (16) and (17), respectively. And the centroid and distance in Equations (12)–(15) will be replaced by (16) and (17).

$$c_i = \frac{w_i}{\|w_i\|}, \quad w_i = \frac{1}{k}\sum_{j=1}^{k} n_j, \tag{16}$$

$$d_{ij} = \left\|p_i - p_j\right\|_2 = \arccos\left(\left\|n_i \cdot n_j\right\|\right). \tag{17}$$

B. EXTERNAL VALIDATION

To calculate the external indices, it needs to get the *true positives* (*TP*), *false positives* (*FP*), *true negatives* (*TN*), and *false negatives* (*FN*). However, region growing only aggregates points with similar attributes into the same cluster and without being able to assign a unique label to each cluster, so the confusion matrix cannot be easily calculated as classification does. An object-based method proposed in the literature [43] was used to compute *TP*, *FP*, and *FN*, which finds a one-to-one correspondence using the maximum or second maximum overlap between extracted clusters and the ground truth. Let the extracted clusters be denoted as $C = \{C_1, C_2, \cdots, C_i\}$, and ground truth of clusters be denoted as $G = \left\{G_1, G_2, \cdots, G_j\right\}$. If C_j shares the maximum overlap with G_j and vice versa, then $\left(C_i, G_j\right)$ is considered as a correspondence. After finding the one-to-one correspondences, *TP*, *FP*, and *TN* were calculated as follows: $TP = C_i \cap G_j$, $FP = C_i \setminus G_j$, and $FN = G_j \setminus C_i$.

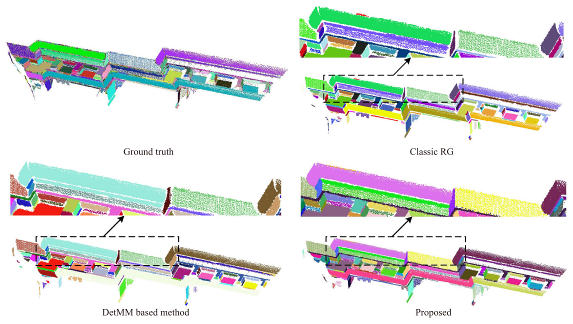

FIGURE 7. Segmentation results with different methods on the data set *region_growing_tutorial*. Parameter setting: Classic RG (classic region growing): σ_{th}=95th, $\theta_{th} = 5°$; DetMM-based method: σ_{th}=95th, $\theta_{th} = 5°$; proposed: $\theta_{th} = 5°$.

Assume κ pairs of one-to-one correspondences have been found. Then, the external indices (*precision*, *recall*, and *F1-score*) can be defined as:

- *Precision* is calculated by $\frac{1}{\kappa}\sum_{i=1}^{\kappa} P_i$, where $P_i = 100 \times \frac{TP_i}{(TP_i+FP_i)}$.
- *Recall* is calculated by $Recall = \frac{1}{\kappa}\sum_{i=1}^{\kappa} R_i$, where $R_i = 100 \times \frac{TP_i}{(TP_i+FN_i)}$;
- *F1-score* is defined as $F1-score = 2 \cdot \frac{Precision \cdot Recall}{(Precision+Recall)}$.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

Two data sets with external labels were used to visualize the segmentation results and analyze the performance of the algorithms: *region_growing_tutorial*, as shown in Figure 7, and *hallway_2*, selected from *Stanford_3d_Dataset_v1.2* [44], as shown in Figure 8. The performance of the proposed method was evaluated in internal and exterior indices and compared with PCA and the DetMM-based method. For all methods, the number of neighbors is set to 50, i.e. $k = 50$, and were run on a laptop with an AMD R7-4800U 4.2 GHz processor.

For the data set *region_growing_tutorial*, the segmentation results of the classic region growing algorithm are shown in the second of Figures 7, and the segmentation results of the DetMM-based method and the proposed method are shown in the third and fourth of Figures 7. Notice that clusters with less than 50 points are not visualized.

For the data set *hallway_2*, the segmentation results of the three methods are shown in Figure 8. It can be seen that the classic region growing could not completely segment the plane (i.e. blank area near edge or corner), due to inaccurate normal estimation in the corner region,as seen in the second of Figure 8. The detMM-based and our methods could

TABLE 2. Internal and external indices calculated by different methods on the data set *region_growing_tutorial* and *hallway_2*.

	index	region_growing_tutoriall			hallway_2		
		[28]	[38]	Proposed	[28]	[38]	Proposed
Internal	CP(↓)	0.2115	0.1061	**0.0903**	0.1118	0.1274	**0.1009**
	SP(↑)	0.5951	0.4139	**0.6337**	0.4324	0.4289	**0.6253**
	DBI(↓)	2.0324	**1.417**	1.7324	1.9665	1.4581	**1.0637**
	DVI(↑)	**10.002**	4.7684	4.9920	**6.7303**	4.7684	4.9174
External	Precision(↑)	100.0	78.4	*82.6*	51.9	90.7	*90.8*
	Recall(↑)	49.3	100.0	*100.0*	89.6	68.3	*100.0*
	F1-score(↑)	66.0	87.9	**90.5**	62.9	77.9	**95.2**

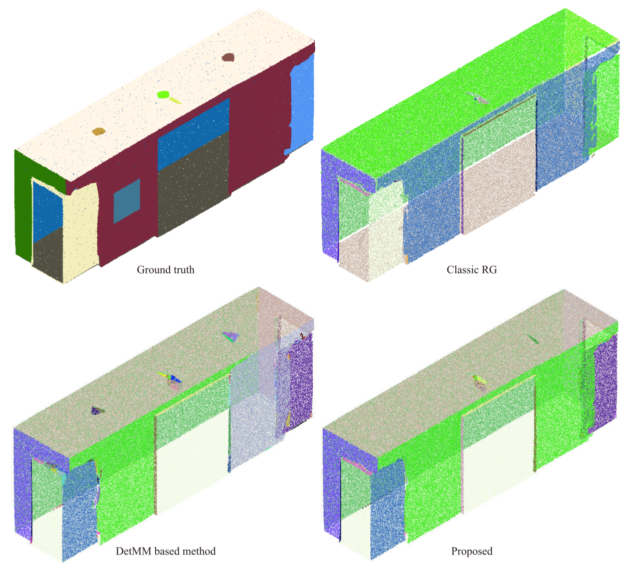

FIGURE 8. Segmentation results with different methods on the data set ***region_growing_tutorial*.** Parameter setting: Classic RG (classic region growing): σ_{th}=95th, θ_{th} = 5°; DetMM-based method: σ_{th}=95th, θ_{th} = 5°; proposed: θ_{th} = 5°.

segment as many plane points as theoretically possible, while the former resulted in over-segmentation at the same angle constraint (i.e. $\theta_{th} = 5°$), as seen in the third and fourth of Figure 8.

Intuitively, it can be seen that the proposed method segmented more points near the edge or corner areas into proper clusters than the other two methods. Quantitatively, internal and external indices were calculated for the three methods on these data sets. Table 2 lists the indices obtained on the data sets *region_growing_tutorial* and *hallway_2*, where ↑ implies that the larger the corresponding index, the better the clustering result, while ↓ denotes the opposite case.

As shown in Table 2, the proposed method performed the best in at least three out of four internal indices and *F1-score*. However, the external indices for *region_growing_tutorial* were lower than those for *hallway_2*, due to the relatively smoother edges and corners. Different values of θ_{th} will result

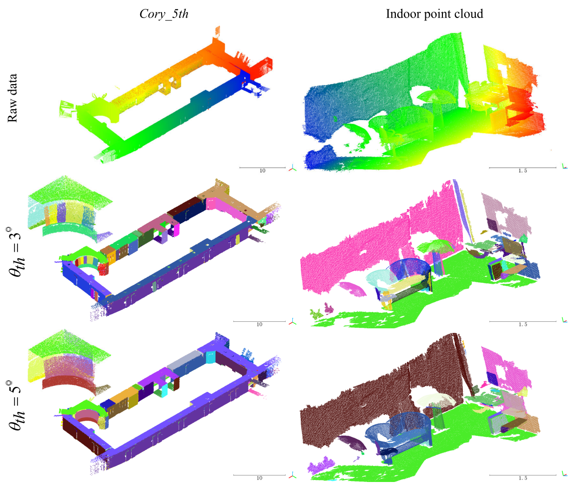

FIGURE 9. Segmentation results on a large data set ***Cory_5th*** and point cloud obtained by Kinect V2 with different values of θ_{th}.

in the different linkages of clusters and, once two clusters are merged, the external indices will be greatly reduced. Overall, the proposed method achieved better performance on the same data set than the traditional region growing algorithm and DetMM-based method.

In addition, two point cloud data sets were used to test the effectiveness and stability of the proposed method: A large point cloud *Cory_5th*, which was captured in the hallways of the 5th floor of Cory Hall on UC Berkeley campus; and an indoor point cloud, which was captured by an RGB-D camera (Kinect v2), as shown in the first row of Figure 9. With the different value of θ_{th}, the segmentation results on the two data sets are shown in the second and third rows of Figure 9. It can be seen that the desired segmentation results were obtained when $\theta_{th} = 5°$, the third row of Figure 9, and both plane and regular curved surfaces were segmented properly in this case.

Finally, the time consumed by the three methods to segment the four point cloud data sets are listed in Table 3. It can be seen that the DetMM-based method required a tremendous amount of time when processing a large-scale point cloud, while the classic region growing algorithm was

TABLE 3. Time consumed by three different methods to segment the four data sets (in minutes).

Data set	Points	Classic RG	DetMM	Proposed
region_growing_tutorial	108,104	0.06	30.96	**1.92**
hallway_2	579,313	0.33	165.92	**3.06**
Cory_5th	1,097,901	0.42	314.44	**9.84**
Indoor point cloud	968,520	0.56	277.39	**7.75**

the least time-consuming but was also inaccurate. The proposed method saved a significant amount of time, compared with the DetMM-based method, although it took much more time than the classic region growing algorithm. This can be considered worthwhile, as we obtained accurate normals and segmentation within a few minutes, even for a point cloud with up to a million points (*Cory_5th*).

V. CONCLUSION

In this paper, we aimed to segment the plane and regular curved surfaces in an indoor point cloud. For this purpose, an improved region growing algorithm was proposed, which consists of two main steps: Accurate normal estimation and strengthened region growing. First, in order to obtain precise normals within an acceptable time, PCA and robust estimator-based PCA were adopted to find the initial normals and refine the SFP normals. Then, the classical region growing algorithm was improved, through setting σ_{th} as the 95th percentile of curvature and strengthening the region growing criteria through connective and coplanar analysis. Based on this model, the desired segmentation effect can be obtained by tuning θ_{th}. After clustering points into groups, labels can be added to each group so that the computer can understand the environment at the semantic level.

The PCA method can calculate point normals rapidly but provides inaccurate results, thus significantly affecting the region growing algorithm which greatly depends on the accuracy of normal estimation. To accurately estimate point normals and reduce time consumption, the method proposed in this paper strikes a balance between accuracy and time consumption. More accurate normals can be obtained with acceptable time consumption. Due to the curvature constraint σ_{th} being fixed as the 95th percentile, the desired segmentation effect could be obtained by tuning only the threshold θ_{th}. Meanwhile, non-connective and non-coplanar points are detected and ignored when carrying out region growing.

To evaluate the performance of the proposed method, internal and external validation indices were calculated, in order to compare the performances of different methods. On two data sets with external labels, the proposed method outperformed a classic region growing method and an advanced DetMM-based method. Finally, two data sets without labels were introduced to test the robustness and effectiveness of the proposed method. In particular, the proposed method could effectively segment both plane and regular curved surface primitives.

However, the proposed method was realized by programming with MATLAB, and no acceleration strategies were adopted. Thus, the time consumption may be further reduced. In addition, it can not segment sampling points on small objects. Finally, the proposed method does not take the density features of the point cloud into account.

In conclusion, the proposed method presented several advantages for point cloud segmentation, regardless of whether planar or curved surfaces are considered. The more accurate normals estimated within an acceptable amount of time can be used in many other application, not just as a pre-requisite for the region growing algorithm.

DATA AVAILABILITY STATEMENT

Publicly available datasets were analyzed in this study. The datasets can be found here: *region_growing_tutorial*: https://raw.github.com/PointCloudLibrary/data/master/tutorials/region_growing_tutorial.pcd (accessed on 26 October 2022), *Stanford_3d_Dataset_v1.2*: http://buildingparser.stanford.edu/dataset.html#Download (accessed on 26 October 2022) and *Cory_5th*: http://www-video.eecs.berkeley.edu/research/indoor/ (accessed on 26 October 2022)

CONFLICT OF INTERESTS

The authors declare no conflict of interest regarding the publication of this article.

ACKNOWLEDGMENT

The authors are grateful to the researchers for providing the public datasets. The authors also thank all the editor and reviewers for their professional comments, which significantly improved this work.

REFERENCES

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016, doi: 10.1109/TRO.2016.2624754.

[2] S. Ochmann, R. Vock, R. Wessel, and R. Klein, "Automatic reconstruction of parametric building models from indoor point clouds," *Comput. Graph.*, vol. 54, pp. 94–103, Feb. 2016, doi: 10.1016/j.cag.2015.07.008.

[3] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," in *Proc. Robot., Sci. Syst.*, Berkeley, CA, USA, 2014, vol. 2, no. 9, pp. 1–9, doi: 10.15607/RSS.2014.X.007.

[4] M. Labbé and F. Michaud, "RTAB-map as an open-source LiDAR and visual simultaneous localization and mapping library for large-scale and long-term online operation," *J. Field Robot.*, vol. 36, no. 2, pp. 416–446, Mar. 2019, doi: 10.1002/rob.21831.

[5] K. Koide, J. Miura, and E. Menegatti, "A portable three-dimensional LiDAR-based system for long-term and wide-area people behavior measurement," *Int. J. Adv. Robot. Syst.*, vol. 16, no. 2, pp. 1–16, 2019, doi: 10.1177/1729881419841532.

[6] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "Kinectfusion: Real-time 3D reconstruction and interaction using a moving depth camera," in *Proc. 24th Annu. ACM Symp. User Interface Softw. Technol.*, Santa Barbara, CA, USA, Oct. 2011, pp. 550–568, doi: 10.1145/2047196.2047270.

[7] T. Whelan, S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison, "ElasticFusion: Dense SLAM without a pose graph," in *Proc. Robot., Sci. Syst.*, Rome, Italy, Jul. 2015, pp. 1–9, doi: 10.15607/RSS.2015.XI.001.

[8] Y. Yang, H. Fang, Y. Fang, and S. Shi, "Three-dimensional point cloud data subtle feature extraction algorithm for laser scanning measurement of large-scale irregular surface in reverse engineering," *Measurement*, vol. 151, Feb. 2020, Art. no. 107220, doi: 10.1016/j.measurement.2019.107220.

[9] X.-F. Han, X.-Y. Yan, and S.-J. Sun, "Novel methods for noisy 3D point cloud based object recognition," *Multimedia Tools Appl.*, vol. 80, no. 17, pp. 26121–26143, Apr. 2021, doi: 10.1007/s11042-021-10794-3.

[10] B. Mahmood, S. Han, and D.-E. Lee, "BIM-based registration and localization of 3D point clouds of indoor scenes using geometric features for augmented reality," *Remote Sens.*, vol. 12, no. 14, p. 2302, Jul. 2020, doi: 10.3390/rs12142302.

[11] Tashi, A. Ullah, M. Watanabe, and A. Kubo, "Analytical point-cloud based geometric modeling for additive manufacturing and its application to cultural heritage preservation," *Appl. Sci.*, vol. 8, no. 5, p. 656, Apr. 2018, doi: 10.3390/app8050656.

[12] E. Grilli, F. Menna, and F. Remondino, "A review of point clouds segmentation and classification algorithms," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 42, p. 339, Apr. 2017, doi: 10.5194/isprs-archives-XLII-2-W3-339-2017.

[13] A. D. Sappa and M. Devy, "Fast range image segmentation by an edge detection strategy," in *Proc. 3rd Int. Conf. 3-D Digit. Imag. Modeling*, 2001, pp. 292–299, doi: 10.1109/IM.2001.924460.

[14] X. Zhang, G. Li, Y. Xiong, and F. He, "3D mesh segmentation using mean-shifted curvature," in *Proc. Int. Conf. Geometric Modeling Process.*, Hangzhou, China, Apr. 2008, pp. 465–474, doi: 10.1007/978-3-540-79246-8_35.

[15] M. Tonini and A. Abellan, "Rockfall detection from terrestrial LiDAR point clouds: A clustering approach using R," *J. Spatial Inf. Sci.*, no. 8, pp. 95–110, 2014. [Online]. Available: https://josis.org/index.php/josis/issue/view/8, doi: 10.5311/JOSIS.2014.8.123.

[16] N. Chehata and F. Bretar, "Terrain modeling from data: Hierarchical K-means filtering and regularization," in *Proc. 15th IEEE Int. Conf. Image Process.*, San Diego, CA, USA, Oct. 2008, pp. 1900–1903, doi: 10.1109/ICIP.2008.4712151.

[17] X. Chen, Q. An, X. Han, Y. Ban, and L. Li, "Control of distributed segmentation of indoor point cloud via homogenization clustering network," *J. Franklin Inst.*, Dec. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0016003221007080, doi: 10.1016/j.jfranklin.2021.12.001.

[18] Z. Luo, Z. Xie, J. Wan, Z. Zeng, L. Liu, and L. Tao, "Indoor 3D point cloud segmentation based on multi-constraint graph clustering," *Remote Sens.*, vol. 15, no. 1, p. 131, Dec. 2022, doi: 10.3390/rs15010131.

[19] D. S. Chen, "A data-driven intermediate level feature extraction algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 7, pp. 749–758, Jul. 1989, doi: 10.1109/34.192470.

[20] P. J. Besl and R. C. Jain, "Segmentation through variable-order surface fitting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, no. 2, pp. 167–192, Mar. 1988, doi: 10.1109/34.3881.

[21] R. Geibel and U. Stilla, "Segmentation of laser altimeter data for building reconstruction: Different procedures and comparison," *Int. Arch. Photogramm. Remote Sens.*, vol. 33, no. 3, pp. 326–334, 2000. [Online]. Available: https://www.isprs.org/proceedings/XXXIII/congress/part3/326_XXXIII-part3.pdf

[22] X. Ning, X. Zhang, Y. Wang, and M. Jaeger, "Segmentation of architecture shape information from 3D point cloud," in *Proc. 8th Int. Conf. Virtual Reality Continuum Appl. Ind.*, Yokohama, Japan, no. 6, Dec. 2009, pp. 127–132, doi: 10.1145/1670252.1670280.

[23] A. Nurunnabi, D. Belton, and G. West, "Robust segmentation in laser scanning 3D point cloud data," in *Proc. DICTA*, 2012, pp. 1–8, doi: 10.1109/DICTA.2012.6411672.

[24] Z. Dong, B. Yang, P. Hu, and S. Scherer, "An efficient global energy optimization approach for robust 3D plane segmentation of point clouds," *ISPRS J. Photogramm. Remote Sens.*, vol. 137, pp. 112–133, Mar. 2018, doi: 10.1016/j.isprsjprs.2018.01.013.

[25] T. Rabbani, F. Van Den Heuvel, and G. Vosselmann, "Segmentation of point clouds using smoothness constraint," *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. 36, no. 5, pp. 248–253, 2006. [Online]. Available: https://www.isprs.org/proceedings/XXXVI/part5/paper/RABB_639.pdf

[26] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *Proc. ICRA*, Shanghai, China, May 2011, pp. 1–4, doi: 10.1109/ICRA.2011.5980567.

[27] J. H. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. Math. Softw.*, vol. 3, no. 3, pp. 209–226, 1977, doi: 10.1145/355744.355745.

[28] G. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," in *Proc. 19th Annu. Conf. Comput. Graph. Interact. Techn.*, Chicago, IL, USA, Jul. 1992, pp. 27–31, doi: 10.1145/133994.134011.

[29] M. Pauly, M. Gross, and L. P. Kobbelt, "Efficient simplification of point-sampled surfaces," in *Proc. IEEE VIS*, Boston, MA, USA, Oct. 2002, pp. 163–170, doi: 10.1109/VISUAL.2002.1183771.

[30] J. Sanchez, F. Denis, D. Coeurjolly, F. Dupont, L. Trassoudaine, and P. Checchin, "Robust normal vector estimation in 3D point clouds through iterative principal component analysis," *ISPRS J. Photogramm. Remote Sens.*, vol. 163, pp. 18–35, May 2020, doi: 10.1016/j.isprsjprs.2020.02.018.

[31] P. J. Huber, "Robust estimation of a location parameter," *Ann. Math. Statist.*, vol. 35, no. 1, pp. 73–101, 1964, doi: 10.1214/aoms/1177703732.

[32] R. He, B. Hu, X. Yuan, and L. Wang, "M-estimators and half-quadratic minimization," in *Robust Recognition via Information Theoretic Learning* (SpringerBriefs in Computer Science), 1st ed. Cham, Switzerland: Springer, 2014, ch. 2, pp. 3–11, doi: 10.1007/978-3-319-07416-0_2.

[33] P. J. Rousseeuw, "Multivariate estimation with high breakdown point," *Math. Stat. Appl.*, vol. 8, no. 37, pp. 283–297, 1985, doi: 10.1007/978-94-009-5438-0_20.

[34] P. J. Rousseeuw and K. Van Driessen, "A fast algorithm for the minimum covariance determinant estimator," *Technometrics*, vol. 41, no. 3, pp. 212–223, 1999, doi: 10.2307/1270566.

[35] F. Cazals and M. Pouget, "Estimating differential quantities using polynomial fitting of osculating jets," *Comput. Aided Geometric Des.*, vol. 22, no. 2, pp. 121–146, Feb. 2005, doi: 10.1016/j.cagd.2004.09.004.

[36] M. Hubert, P. J. Rousseeuw, and T. Verdonck, "A deterministic algorithm for robust location and scatter," *J. Comput. Graph. Statist.*, vol. 21, no. 3, pp. 618–637, Jul. 2012, doi: 10.1080/10618600.2012.672100.

[37] M. Hubert, P. Rousseeuw, D. Vanpaemel, and T. Verdonck, "The DetS and DetMM estimators for multivariate location and scatter," *Comput. Statist. Data Anal.*, vol. 81, pp. 64–75, Jan. 2015, doi: 10.1016/j.csda.2014.07.013.

[38] A. Khaloo and D. Lattanzi, "Robust normal estimation and region growing segmentation of infrastructure 3D point cloud models," *Adv. Eng. Informat.*, vol. 34, pp. 1–16, Oct. 2017, doi: 10.1016/j.aei.2017.07.002.

[39] P. J. Rousseeuw and C. Croux, "Alternatives to the median absolute deviation," *J. Amer. Statist. Assoc.*, vol. 88, no. 424, pp. 1273–1283, Dec. 1993, doi: 10.1080/01621459.1993.10476408.

[40] M. Wei, J. Yu, W.-M. Pang, J. Wang, J. Qin, L. Liu, and P.-A. Heng, "Binormal filtering for mesh denoising," *IEEE Trans. Vis. Comput. Graphics*, vol. 21, no. 1, pp. 43–55, Jan. 2015, doi: 10.1109/TVCG.2014.2326872.

[41] H. Son, C. Kim, and C. Kim, "Fully automated as-built 3D pipeline extraction method from laser-scanned data based on curvature computation," *J. Comput. Civil Eng.*, vol. 29, no. 4, 2015, Art. no. B4014003, doi: 10.1061/(ASCE)CP.1943-5487.0000401.

[42] A. Al-Rawabdeh, F. He, and A. Habib, "Automated feature-based down-sampling approaches for fine registration of irregular point clouds," *Remote Sens.*, vol. 12, no. 7, p. 1224, Apr. 2020, doi: 10.3390/rs12071224.

[43] M. Awrangjeb and C. S. Fraser, "An automatic and threshold-free performance evaluation system for building extraction techniques from airborne LiDAR data," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 10, pp. 4184–4198, Oct. 2014, doi: 10.1109/JSTARS.2014.2318694.

[44] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, "3D semantic parsing of large-scale indoor spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 1534–1543, doi: 10.1109/CVPR.2016.170.

WEI WANG received the B.S. and M.S. degrees in electronic engineering from Northwest Normal University, China. He is currently pursuing the Ph.D. degree in pattern recognition and knowledge discovery with the Chongqing University of Posts and Telecommunications. His research interests include point cloud data processing, mobile robot navigation, machine learning, and pattern recognition.

YI ZHANG received the Ph.D. degree in mechanical manufacturing and automation from the Huazhong University of Science and Technology, Wuhan, China. He completed his post-doctoral training in intelligent multimode human–computer interaction with the University of Essex, London, U.K. He is currently a Professor and a Doctoral Supervisor with the Advanced Manufacturing Engineering College, Chongqing University of Posts and Telecommunications. His research interests include robot automatic control and human–computer interaction.

YANG WANG received the M.S. degree in computer technology from the Kunming University of Science and Technology, China. She is currently pursuing the Ph.D. degree in pattern recognition and knowledge discovery with the Chongqing University of Posts and Telecommunications. Her research interests include machine learning, pattern recognition, mobile robot, and VSLAM.

GENGYU GE received the B.S. degree in information and computing science from Chuzhou University, Chuzhou, China, in 2011, and the M.S. degree in computer system architecture from Southwest University, Chongqing, China, in 2014. He is currently pursuing the Ph.D. degree with the Chongqing University of Posts and Telecommunications. His research interests include mobile robot navigation, semantic slam, and embedded system application.

QIN JIANG received the B.S. and M.S. degrees in biomedical engineering from the Chongqing University of Technology, China. She is currently pursuing the Ph.D. degree in pattern recognition and knowledge discovery with the Chongqing University of Posts and Telecommunications. Her research interests include machine learning, pattern recognition, biomedical signal process, and brain–computer interface.

LIHE HU received the M.S. degree in mechanical engineering from Chongqing Jiao Tong University, China, in 2018. He is currently pursuing the Ph.D. degree with the China EU Intelligent System and Robot Joint Laboratory, Chongqing University of Posts and Telecommunications. His research interests include mobile robot semantic mapping, real time mapping, semantic slam, semantic segmentation, and deep learning.