Queries to the Author

When you submit your corrections, please either annotate the IEEE Proof PDF or send a list of corrections. Do not send new source files as we do not reconvert them at this production stage.

Authors: Carefully check the page proofs (and coordinate with all authors); additional changes or updates WILL NOT be accepted after the article is published online/print in its final form. Please check author names and affiliations, funding, as well as the overall article for any errors prior to sending in your author proof corrections. Your article has been peer reviewed, accepted as final, and sent in to IEEE. No text changes have been made to the main part of the article as dictated by the editorial level of service for your publication.

Plane Segmentation Based on the Optimal-Vector-Field in LiDAR Point Clouds

Sheng Xu, Ruisheng Wang, Hao Wang, and Ruigang Yang, *Senior Member, IEEE*

Abstract—One key challenge in the point cloud segmentation is the detection and split of overlapping regions between different planes. The existing methods depend on the similarity and the dissimilarity in neighbor regions without a global constraint, which brings the 'over-' and 'under-' segmentation in the results. Hence, this paper presents a pipeline of the accurate plane segmentation for point clouds to address the shortcoming in the local optimization. There are two phases included in the proposed segmentation process. One is a local phase to calculate connectivity scores between different planes based on local variations of surface normals. In this phase, a new optimal-vector-field is formulated to detect the plane intersections. The optimal-vector-field is large in magnitude at plane intersections and vanishing at other regions. The other one is a global phase to smooth local segmentation cues to mimic leading eigenvector computation in the graph-cut. Evaluation of two datasets shows that the achieved precision and recall is 94.50 percent and 90.81 percent on the collected mobile LiDAR data and obtains an average accuracy of 75.4 percent on an open benchmark, which outperforms the state-of-the-art methods in terms of completeness and correctness.

Index Terms—Plane segmentation, optimal-vector-field, point clouds, surface normals, graph-cut

1 Introduction

Segmentation from point clouds is the process of partitioning input data into multiple regions, which has provided benefits for many applications, e.g., artifact modeling [1], vegetation reconstruction [2] and ROI (Region of Interest) detection [3]. In general, segmentation results can be divided into two levels, the primitive level, e.g., line [4], plane [5] and cylinder [6], and individual object level [7], [8]. A new commonly used approach for the object-level segmentation is based on the deep learning [9], [10], which is a supervised learning technique. However, the definition of an individual object is related to a definite application, e.g., the vehicle detection [10] or semantic scenes segmentation [9]. Besides, the supervised learning method needs to manually segment a large number of objects for setting the training set, which is often tedious, redundant in the complex outdoor scene, and requires a high-performance GPU (Graphics Processing Unit) for accelerating algorithms. Therefore, this paper proposes a general unsupervised plane segmentation method to deal with the existing segmentation bottleneck, i.e., the incorrect split of overlapping regions from different planes, and to demonstrate the comparison of results with deep learning methods.

Contributions of the proposed segmentation pipeline lie in a local phase and a global phase. The local phase calculates the connectivity scores of planes based on the local variations of surface normals. The global phase performs the leading eigenvector computation to produce the desired segmentation. Two key points of this paper are as follows.

1) We optimize a new optimal-vector-field to provide local segmentation cues, which is large in magnitude at plane intersections and vanishing at other regions, for obtaining segmentation cues.
2) In the existing work, the optimal segmentation is often based on an object division strategy. For example, in the graph-cut process, users have to define the background and foreground objects to split two regions. The multi-object segmentation requires users to iteratively use a single-cut or design a multi-way cut for the optimization, which involves much calculation. To address this issue, we propose a non-iterative strategy for the accurate plane segmentation by performing a single graph-cut on the obtained cues.

- *S. Xu is with the College of Information Science and Technology, College of Landscape Architecture, Nanjing Forestry University, Nanjing, Jiangsu 210037, China. E-mail: sheng.xu2@ucalgary.ca.*
- *R. Wang is with the School of Geographical Sciences, Guangzhou University, Guangzhou, Guangdong 510006, China, and also with the Department of Geomatics Engineering, University of Calgary, Calgary, AB T2N 1N4, Canada. E-mail: ruiswang@ucalgary.ca.*
- *H. Wang is with the College of Landscape Architecture, Nanjing Forestry University, Nanjing, Jiangsu 210037, China. E-mail: wh9816@126.com.*
- *R. Yang is with the Computer Science, University of Kentucky, Lexington, KY 40507. E-mail: ryang@cs.uky.edu.*

Manuscript received 8 Feb. 2020; revised 9 May 2020; accepted 11 May 2020. Date of publication 0 . 0000; date of current version 0 . 0000.
(Corresponding author: Ruisheng Wang.)
Recommended for acceptance by Y. Furukawa.
Digital Object Identifier no. 10.1109/TPAMI.2020.2994935

2 Related Work

Nowadays, researchers have developed various methods to fit planes, such as building facades or walls. Planes are usually detected with the region growing technique [11], which increases object regions from given seed positions and stops the propagation based on user-defined priors. For example, [12] merge points sharing similar normal vectors and [13] group points which can be fitted by the same plane function. Although region growingbased algorithms output high quality and consistent models of planes, they fail to segment those incomplete or occluded planes.

In order to be robust to the incomplete input data, [14] and [15] introduce the Random Sample Consensus (RANSAC) to segment points based on their distance to primitive models, e.g., spheres, cylinders, and cones. RANSAC-detected planes can be hierarchically assembled, and the discovery of their intersections helps recover full planes. RANSAC-based methods work well in environments mainly made of planar surfaces, which are usually followed by a clustering of the parameter space to refine segmentation results. For example, the k-means approach [16], [17] partition points into different sets by ensuring that the sum of the distance of each point in the cluster to the center achieves its minimum. In the work of [18], they propose a plane extraction method based on an agglomerative clustering (PEAC). They segment planes from point clouds efficiently when the input point clouds are well-organized. The segmentation accuracy of the above-mentioned approaches highly depends on parameters. Their results are more likely to be locally optimal, resulting in a high under- or over-segmentation rate when choosing non-optimal parameters.

A well-known globally optimal segmentation technique is the graph-based method, which treats the point cloud segmentation as a labeling problem to achieve the minimization. Each point is assigned with one possible label and the challenge is to find the optimal label configuration for all points according to energy functions. Two prominent examples are the normalized-cut [19] and the graph-cut [20], which build a graph that formulates and smooths local segmentation cues to produce the desired segmentation. The normalized-cut has been used in 2D image segmentation for a long time [21], [22]. It partitions a graph into two disjoint groups by minimizing the dissimilarity within each group and maximizing the dissimilarity between different groups. In the point cloud segmentation, [7] achieve high accuracy in the extraction of pole-like objects from mobile laser scanning (MLS) data and [23], [24] succeed in reducing the rate of the over-segmentation. The solution cut for separating the graph into two optimal parts is obtained in a similar way as in 2D segmentation after adding the elevation information. The normalized-cut requires users to initialize the number of objects in the multi-target segmentation and uses the one-vs-others strategy to iteratively partition scene into two disjoint groups. The graph-cut is proved as another efficient interactive segmentation method for natural images as shown in [25], [26]. The authors formulate the energy function using binary variables, and values (i.e., 0 or 1) indicate whether a pixel belongs to the foreground or background. The solution cut for separating the graph into the optimal background and foreground is obtained by solving the minimum cut of the graph. The graph-cut has obtained an impressive performance in the segmentation of LiDAR point clouds as shown in the work of [27] and [28]. However, graph-cut usually requires a computer-human interaction step to indicate the foreground and background. Moreover, similar to the normalized-cut, graph-cut also chooses the one-vs-others strategy to iteratively detect the foreground and background in multi-target segmentation. The one-vs-others strategy degrades the superiority of the algorithms for the binary labeling segmentation, e.g., the requirement of manually initializing the

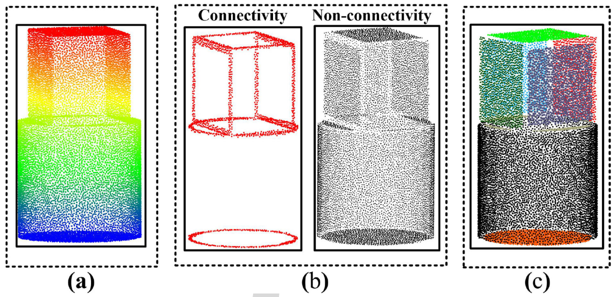


Fig. 1. The process of the proposed plane segmentation. (a) Input scene. (b) The divided connectivity and non-connectivity regions. (c) The plane segmentation result. Distinct colors mean different planes.

number of labels and the propagation of segmentation errors in each iteration.

Before the discussion of the plane segmentation, we know that there are two regions in a point cloud scene. One is the connectivity region containing points from plane intersections. The other one is the non-connectivity region containing the rest of the input points. Since connectivity regions consist of all intersections, if one removes all connectivity regions, the input scene will be split into disjoint clusters and each cluster stands for a plane. However, two different surfaces can be consistent with the semantic structure. For two touching surfaces, there is a high chance to be clustered as one set. Therefore, a robust indicator to detect plane intersections is necessary, which will be implemented by a new optimal-vector-field. The following is a brief overview of the proposed segmentation. In the local phase, we formulate a new optimal-vector-field to detect the potential intersections in point clouds. Then, in the global phase, we divide the input scene into connectivity and non-connectivity regions by using a graph-based segmentation model. After those two phases, we cluster points from non-connectivity regions into disjoint groups. Fig. 1 demonstrates an example of our segmentation. Fig. 1a shows the input scene, (b) shows the result of the binary division, and (c) shows the result of the plane segmentation.

3 Normal Vector Estimation and Connectivity Value Calculation

In our work, the optimal-vector-field is defined as an assignment of a vector to each point in a subset of space based on the normal vector estimation. The normal vector at a point is approximated as the normal to the surface estimated by its k-nearest neighborhood points. Assuming that there are k points in the estimation, based on singular value decomposition (SVD) method we have

$$\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \dots & \dots & \dots \\ x_k & y_k & z_k \end{bmatrix} = \mathbf{D}_{k\times 3} = \mathbf{U}_{k\times k}\mathbf{S}_{k\times 3}\mathbf{V}^{\top}_{3\times 3}, \tag{1}$$

where $\mathbf{D}$ is the input matrix decomposed into the matrices $\mathbf{U}$, $\mathbf{S}$ and $\mathbf{V}$. The column vector in $\mathbf{V}$, which corresponds to the smallest eigenvalue in the decomposition (usually the last one), is chosen as the normal vector at the given point. Fig. 2 shows the calculated normal vectors of a point cloud

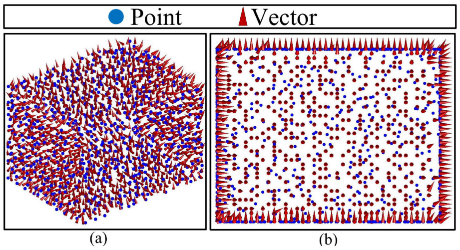


Fig. 2. Visualization of normal vectors. (a) Oblique view. (b) Top view.

in two different views. Each point is attached to a normal vector visualized as a cone.

One key issue of the normal vector estimation at points is their inconsistent directions, i.e., vectors are in the reverse direction on a local plane. The commonly used solution is by measuring the difference of neighboring vectors and propagating the consistent orientation along a surface [29]. Our idea is that, for a current point c_0 and its neighboring points c_1, c_2 and c_3 as shown in Fig. 3a, if both $|cos(\angle(\mathbf{V}(c_0), \overrightarrow{c_0, c_j}))|$ and $|cos(\angle(\mathbf{V}(c_j), \overrightarrow{c_0, c_j}))|$ exceed a threshold (0.9 in our case), we flip the normal direction at c_j, i.e., c_2 in this example. Consistency results are shown in Fig. 3b. To ensure the convergence, first, we split the input scene into proper disjoint cubes (0.5 m by 0.5 m by 0.5 m in this work) before the flipping operation. Then, for each cube, we randomly select one point as c_0 and conduct the point-by-point consistency operation for other points in this cube based on the angle information.

Next, we show the calculation of connectivity scores of points to obtain the magnitude of each point in the optimal-vector-field. In our work, the connectivity score $h(x, y, z)$ of a point c_i at the coordinate (x, y, z) is calculated by

$$h(x, y, z) = \frac{1}{\sum_{c_j \in c_i^s} |\mathbf{V}(c_i) \cdot \mathbf{V}(c_j)|}, \quad (2)$$

where $\mathbf{V}(c_i)$ is the normal vector at c_i, and c_i^s is the set of c_i's k-nearest neighbors. Based on Eq. (2), since a point from the non-connectivity region sharing the similar normal vector with its neighborhood points, there will be a much smaller h than a point from the connectivity region. The illustration of connectivity scores is shown in Fig. 4. Connectivity scores of points from the intersection of two planes are much higher than points within a plane as shown in Fig. 4a.

4 Optimal-Vector-Field Optimization

Although the connectivity score defined in Eq. (2) can highlight the region of connectivity between planes as shown in

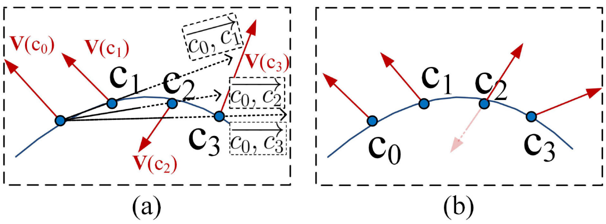


Fig. 3. Consistency of normal vectors based on the angle information. (a) Initial normal vectors and points. (b) The adjustment of the normal vector at c_2.

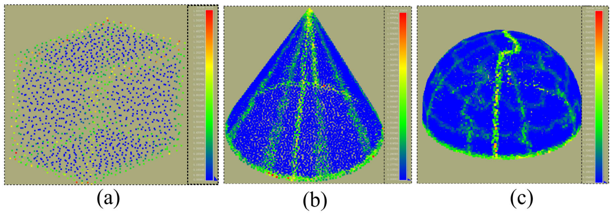


Fig. 4. Illustration of connectivity scores of point clouds. (a) Point clouds of a cube. (b) Point clouds of a cone. (c) Point clouds of a half-sphere.

Fig. 4a, the score is difficult to indicate the connectivity in complex surfaces as shown in Figs. 4b and 4c. The normal vectors at points from those surfaces are in different directions, and the score depends on the curvature information. To address this issue, we take the gradient of points into consideration and propose an optimal-vector-field procedure for obtaining segmentation cues using the global information. The optimization is shown in Fig. 5. In the vector field modeling, the gradient information and connectivity score are calculated to formulate the objective function. To solve the energy function, both the analytical solution and the numerical solution are derived. To ensure the convergence, the stableness of the solution is analyzed at the end of the procedure.

The key to the above-mentioned procedure is formulation and optimization of the optimal-vector-field, whose direction is intended to be consistent with the normal vector and the magnitude will be related to the connectivity score. We design three principles for the optimal-vector-field formulation. First, the vector direction is perpendicular to the surface fitted by its k-neighborhood points. Second, vectors are large in magnitude only at the points from the connectivity regions. Third, vectors are nearly zero in magnitude at the points from the non-connectivity regions. Details of the optimal-vector-field optimization are shown below.

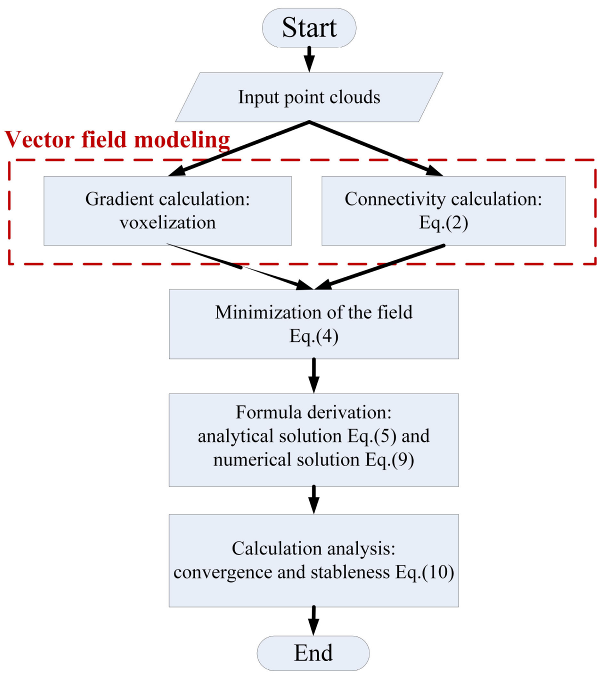


Fig. 5. The algorithm figure to describe the optimal-vector-field optimization procedure.

The optimal-vector-field magnitude F of a point at the coordinate (x, y, z) is formed as a functional

$$F(u, v, w) = [u(x, y, z), v(x, y, z), w(x, y, z)], \tag{3}$$

where u, v and w are functions of the coordinate (x, y, z) and they measure the field in different dimensions. The following equation aims to optimize an energy function which follows the principle that it keeps the field F nearly equal to h in the connectivity region, but forces the field to be slowly-varying in the non-connectivity region. The proposed energy function is defined as

$$\begin{aligned} E &= \iiint \big(\lambda E_g + E_h\big) dxdydz \\ &= \iiint \{\lambda\big(\triangledown u \cdot \triangledown u^\top + \triangledown v \cdot \triangledown v^\top + \triangledown w \cdot \triangledown w^\top\big) \\ &\quad + h \times (F - h)^2\} dxdydz, \end{aligned} \tag{4}$$

where E_g measures the gradient of u, v and w, and E_h detects the connectivity region using the connectivity score h. The target of E_g is to decrease the magnitude of the optimal-vector-field within a plane using the gradient information, and E_h is to highlight the magnitude of the optimal-vector-field using the connectivity score cue. The regularization parameter λ is to balance E_g and E_h in the integrand. If h is small, the energy E is dominated by E_g, which is the sum of the squares of F's partial derivative and tends to be a slowly varying field. If h is large, E_h dominates the energy and E achieves the minimization by setting $F = h$.

The gradient is defined based on the work of [30] using the point's density. At first, voxels are generated for the point cloud. Then, the density at a point is approximated by the number of points in the generated voxel at this point. Finally, the gradient of a point is calculated by the difference of the density at adjacent points. Details of setting the voxel size and number are shown in [30]. No matter the point cloud is uniformly sampled or not, the gradient of points from the intersection of surfaces will be large in more than one direction [30].

In the optimization of Eq. (4), we use Euler equation [31] to minimize the energy. The problem-solving process is shown in Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2020.2994935, and the result is

$$\lambda \triangle u - h \times (u - h) = 0, \tag{5a}$$

$$\lambda \triangle v - h \times (v - h) = 0, \tag{5b}$$

$$\lambda \triangle w - h \times (w - h) = 0, \tag{5c}$$

where $\triangle$ is the Laplace operator.

To calculate Eq. (5) in a numerical method, we add a time t in u, v and w as $u_t(x, y, z, t)$, $v_t(x, y, z, t)$ and $w_t(x, y, z, t)$, respectively. Equation (5) is solved by regarding u, v and w as functions of t and calculated as

$$\begin{aligned} u_t(x, y, z, t) &= \lambda \triangle u(x, y, z, t) - h(x, y, z) \\ &\quad \cdot (u(x, y, z, t) - h(x, y, z)) = 0, \end{aligned} \tag{6a}$$

$$\begin{aligned} v_t(x, y, z, t) &= \lambda \triangle v(x, y, z, t) - h(x, y, z) \\ &\quad \cdot (v(x, y, z, t) - h(x, y, z)) = 0, \end{aligned} \tag{6b}$$

$$\begin{aligned} w_t(x, y, z, t) &= \lambda \triangle w(x, y, z, t) - h(x, y, z) \\ &\quad \cdot (w(x, y, z, t) - h(x, y, z)) = 0. \end{aligned} \tag{6c}$$

Since the above-formulated diffusion equations are decoupled, they can be solved as separate scalar partial differential equations in u, v and w. The steady-state solution of those diffusion equations is the answer to Eq. (4). To set up the iteration, let the spacing between points be Δx, Δy, and Δz and the time step for each iteration be Δt. Then, the required partial derivatives are approximated as

$$u_t = \frac{1}{\Delta t}(u^{t+1}_{x,y,z} - u^t_{x,y,z}), \tag{7a}$$

$$v_t = \frac{1}{\Delta t}(v^{t+1}_{x,y,z} - v^t_{x,y,z}), \tag{7b}$$

$$w_t = \frac{1}{\Delta t}(w^{t+1}_{x,y,z} - w^t_{x,y,z}). \tag{7c}$$

The $\triangle u$ is calculated as

$$\begin{aligned} \triangle u &= \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \\ &= \frac{u_{xx}}{\Delta x \Delta y \Delta z} + \frac{u_{yy}}{\Delta x \Delta y \Delta z} + \frac{u_{zz}}{\Delta x \Delta y \Delta z}, \end{aligned} \tag{8}$$

where

$$\begin{aligned} u_{xx} &= u_{x+1,y,z} + u_{x-1,y,z} - 2u_{x,y,z}, \\ u_{yy} &= u_{x,y+1,z} + u_{x,y-1,z} - 2u_{x,y,z}, \\ u_{zz} &= u_{x,y,z+1} + u_{x,y,z-1} - 2u_{x,y,z}. \end{aligned}$$

Combined Eqs. (6)a, (7)a, and (8), we have

$$\begin{aligned} \frac{1}{\Delta t}\Big(u^{t+1}_{x,y,z} - u^t_{x,y,z}\Big) &= \frac{\lambda(u^t_{xx} + u^t_{yy} + u^t_{zz})}{\Delta x \Delta y \Delta z} \\ &\quad - h_{x,y,z}\Big(u^t_{x,y,z} - h_{x,y,z}\Big). \end{aligned}$$

Rewrite this equation to obtain the update formula as

$$\begin{aligned} u^{t+1}_{x,y,z} &= \big(1 - h_{x,y,z}\Delta t\big)u^t_{x,y,z} \\ &\quad + \frac{\lambda \Delta t}{\Delta x \Delta y \Delta z} \cdot \Big(u^t_{xx} + u^t_{yy} + u^t_{zz}\Big) + \Delta t h^2_{x,y,z}. \end{aligned} \tag{9}$$

Similarly, we can obtain the update formula for $v^{t+1}_{x,y,z}$ and $w^{t+1}_{x,y,z}$.

To ensure that the proposed numerical scheme converges well in the calculation of $u^{t+1}_{x,y,z}$, $v^{t+1}_{x,y,z}$ and $w^{t+1}_{x,y,z}$, it is necessary to analyze the stableness of the proposed scheme. As mentioned in Courant-Friedrichs-Levy condition (CFL condition) [32], the numerical domain of dependence must contain the physical domain of dependence in order to obtain a stable solution. The stability of a scheme means that mistakes at one-time step of the calculation do not increase errors as the computations continue. In Lax equivalence theorem [33], stability is the necessary and sufficient condition for the convergence of a scheme. Our stability analysis is based on the Von Neumann method [34]. Details of the convergence analysis are shown in Appendix B, available in the online supplemental material. When Δx, Δy, Δz and λ are fixed, we find that the following restriction in Eq. (10) on the time-step must be maintained to guarantee convergence of the iterative process in Eq. (9).

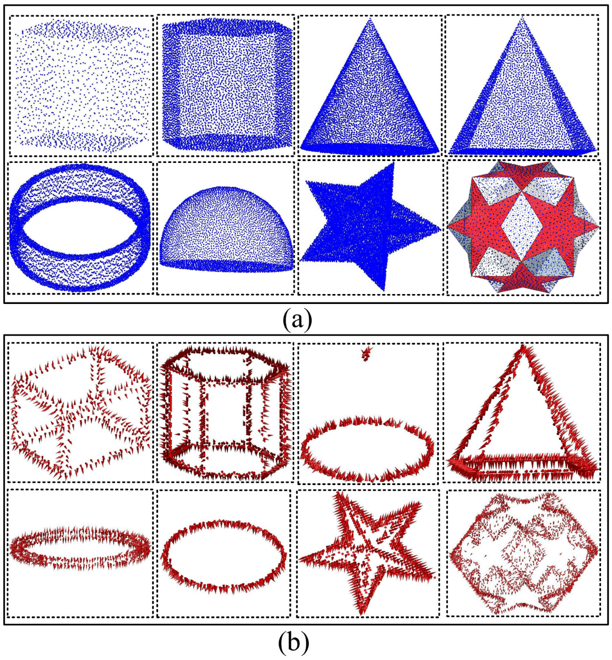


Fig. 6. Illustration of the optimal-vector-field of point clouds. (a) Input datasets. (b) Optimal-vector-field of (a).

$$\Delta t \leq \frac{\Delta x \Delta y \Delta z}{6\lambda}. \quad (10)$$

In the calculation of the optimal-vector-field, when Δx, Δy and Δz are large, the convergence can be made to be fast by choosing a large Δt. F is iteratively calculated by optimizing energy with a diffusion process. F can be regarded as a set of vectors. u, v, and w are the component of F in different dimensions. Those components are calculated through the diffusion process and provide cues to the segmentation in the local phase. Fig. 6 illustrates the optimal-vector-field of different point clouds. Fig. 6a shows eight input point cloud sets, including the cube, cylinder, cone, pyramid, ring, half-sphere, star, and mixed-shape. Fig. 6b displays the corresponding optimal-vector-field in Fig. 6a. In the visualization of the optimal-vector-field, if the magnitude at a point is non-zero, there will be a small red cone as shown in Fig. 6b. Points from the connectivity region have a large magnitude.

5 Plane Segmentation With a Single Graph-Cut

This section aims to divide the input scene into the connectivity and non-connectivity regions by using a single graph-cut. The optimization procedure is shown in Fig. 7. In the graph modeling, nodes are built by voxels and the weight calculation is based on the defined data term and smoothness term. The key to the data term and the smoothness term is the divergence calculation and the curvature distance, respectively. The objective function formulated by the data term and smoothness term will be minimized by a single graph-cut. The foreground and background in the graph-cut process are corresponding to our connectivity regions to be removed and non-connectivity regions to be combined. Details of the graph-cut optimization are shown below.

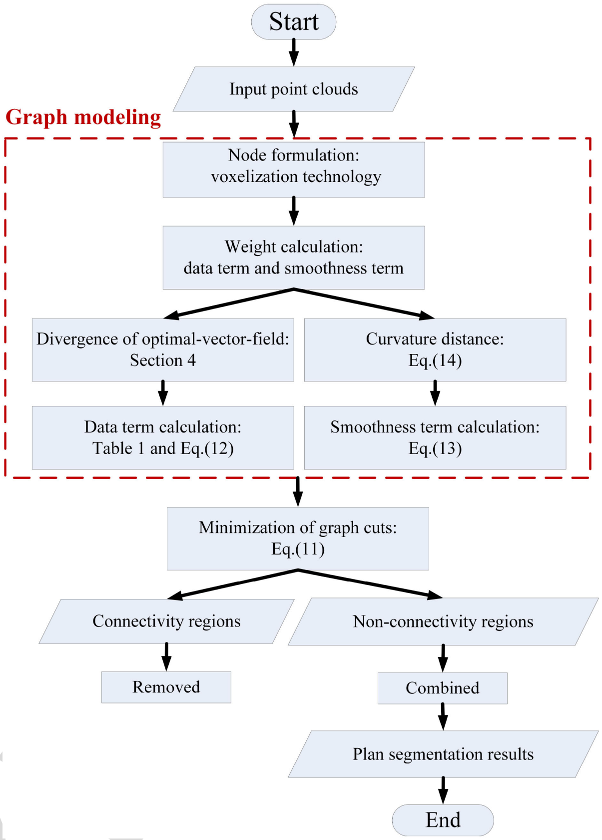


Fig. 7. The algorithm figure to describe the graph-cut optimization procedure.

Assume that C is an arbitrary set of points and N is a set of the neighborhood system to represent all pairs $\{c, c'\}$ of neighboring elements in C. Let $L = (l_1, \ldots, l_i, \ldots, l_{|C|})$ be a binary vector to describe the label configuration of each point for the segmentation. Our segmentation energy function is formulated as

$$E(C, L) = D(C, L) + \gamma S(C, L), \quad (11)$$

where $D(C, L)$ is the data term to measure how appropriate a label l_c is for a point $c \in C$, and $S(C, L)$ is the smoothness term to constrain labels of neighboring points. The coefficient γ is to balance the above two terms.

Based on the property of the optimal-vector-field, we know that points from the connectivity region have a large divergence. Therefore, we formulate the data term as

$$D(C, L) = \sum_{c \in C} d(c, l_c), \quad (12)$$

where $d(c, l_c)$ depends on the divergence of the optimal-vector-field at a point. The binary label l_c is chosen from 0 (the connectivity region) and 1 (the non-connectivity region). Our rule is that $d(c, l_c)$ should be penalized if the divergence $div(c)$ is less than ϵ, when l_c is 0, or if the divergence $div(c)$ is larger than ϵ, when l_c is 1. ϵ is a small positive number. In Table 1, we show penalties for points of different labels based on the divergence.

Next is the formulation of the smoothness term. Since the curvature value at points from the connectivity region is usually much larger than those from the non-connectivity

TABLE 1
The Setting of Penalties in the Data Term Calculation

$div(c)$	l_c=0	l_c=1
$> \epsilon$	$d(c, l_c)$=0	$d(c, l_c)$=1
$\leq \epsilon$	$d(c, l_c)$=1	$d(c, l_c)$=0

region, we penalize the smoothness of labels by the curvedness as

$$S(C, L) = \sum_{\{c,c'\} \in N} \beta_{c,c'} \cdot \delta(l_c, l_{c'}), \tag{13}$$

where N contains all unordered pairs of neighboring points in C. $\delta(l_c, l_{c'})$ outputs a binary number to indicate the continuity of labels, i.e., if $l_c \neq l_{c'}$, $\delta(l_c, l_{c'}) = 1$, otherwise $\delta(l_c, l_{c'}) = 0$. $\beta_{c,c'}$ is interpreted as a penalty for the discontinuity of neighbors' labels and formulated as a function of the curvature distances by

$$\beta_{c,c'} = e^{-(r_c - r_{c'})^2}, \tag{14}$$

where r_c and $r_{c'}$ are the curvedness [35] at the point c and c', respectively. The curvedness is to describe how highly or gently curved a surface is at a point. It is zero only for planar patches and tends to be large for the uneven planes. When $\delta(l_c, l_{c'}) = 1$, if the curvedness at the point c and c is similar, i.e., within the non-connectivity region, $\beta_{c,c'}$ is large; if their curvedness values are different, i.e., within the connectivity region, $\beta_{c,c'}$ is small. Details of the curvedness calculation are shown in [35], which is based on the Gaussian curvature and mean curvature.

The minimization of Eq. (11) in the global phase can be achieved by many developed models, e.g., Laplacian smoothing [36], anisotropic diffusion [37], graph-cut [20] and normalized-cut [19]. Since points from the connectivity region are much less than those from the non-connectivity region, we prefer to choose the graph-cut model to divide a scene into the connectivity and non-connectivity region. In the graph formulation, we use the voxelization technology to divide input point clouds into voxels. The size of voxels depends on the density of point clouds. If the input scene is divided by large voxels, the connectivity regions will become blurred. If it is divided by small voxels, the time-cost will increase greatly. In our case, we fix the size of voxels as 1 cm by 1 cm by 1 cm. Each voxel will be regarded as a node in the graph. Every two nodes are weighted by the formulated data term and smoothness term. If the euclidean distance of two nodes is larger than 1 m, the weight between them will be set as infinite. There are lots of infinities in the weight matrix. To relieve the space complexity, one can use a sparse matrix strategy to store graph nodes. When the graph is built, users can conduct the graph-cut method [38], [39], [40] to divide nodes into background and foreground, i.e., non-connectivity regions and connectivity regions, respectively.

After we remove the connectivity regions of a point cloud, planes will have no intersections. Therefore, the non-connectivity regions can be clustered into a set of disjoint groups based on the euclidean distance easily. Each region

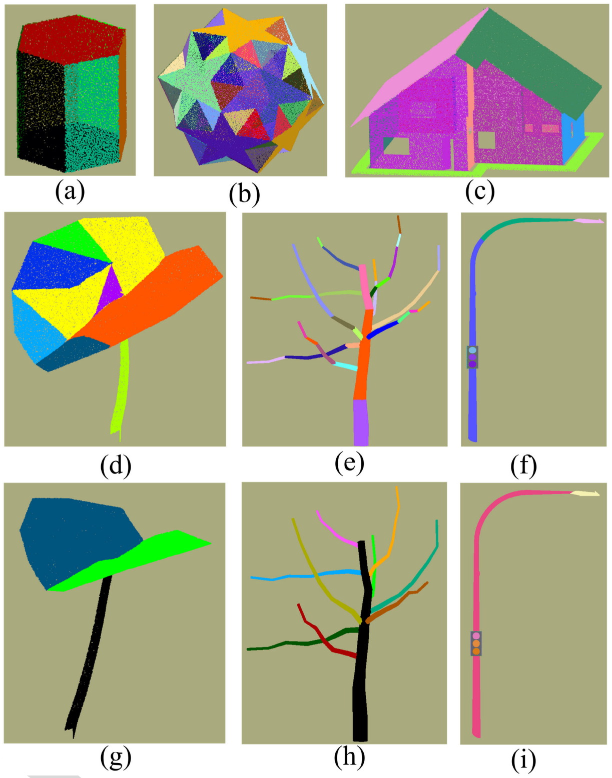


Fig. 8. The effect of the data term and smoothness term in the proposed optimization procedure. (a)-(f) are optimized with the data term only (O-D). (g)-(i) are optimized with both data term and smoothness term (O-DS).

is regarded as a plane. It is worth noting that the points of intersections will not present in the final segmentation.

To help further understand the effect of the optimization with data term (O-D) and the optimization with data term and smoothness term (O-DS) in the plane segmentation, we conduct an ablation study as shown in Fig. 8. The first two rows show the segmentation results by using the O-D strategy. Although the data term segments planes from the synthetic point clouds in Figs. 8a, 8b, and 8c, there appears the over-segmentation in complex planes as shown in Figs. 8d, 8e, and 8f. The leaf is divided into pieces due to the added slight wrinkle. The tree trunk and lamppost are over-split into multiple segments. This is because the data term tends to be sensitive to the bending non-connectivity regions. In the ablation study, there is no difference in results between the O-D and O-DS from scenes in (a) to (c). The advantage of O-DS lies in addressing the above-mentioned over-segmentation issues in bending regions as shown in Figs. 8g, 8h, and 8i.

6 Evaluations and Results

6.1 Comparison of Different Methods

Since the ground-truth for the plane segmentation in outdoor scenes is difficult to be defined and reproduced, especially in vegetation and thin pole-like objects, our ground-truth for the vegetation and poles are achieved by separating the visually independent objects manually. We use the CloudCompare visualization software (www.danielgm.net) to segment each independent object one by one. The manually obtained ground truth (MGT) is regarded as a kind of

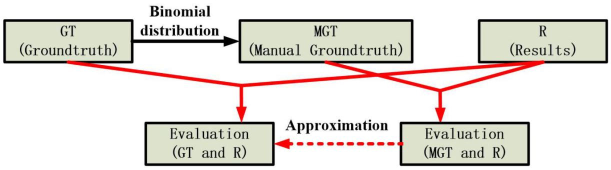


Fig. 9. Approximate evaluation method.

the binomial distribution of ground-truth (GT). Assume that R is the segmentation result from a specific algorithm. From the Bayes rule, we know

$$\begin{aligned} &Pro(R, GT, MGT) \\ &= Pro(R|GT)Pro(GT)Pro(MGT|GT, R) \\ &= Pro(R|MGT)Pro(MGT)Pro(GT|MGT, R). \end{aligned}$$

Therefore,

$$\begin{aligned} &Pro(R|GT) \\ &= \frac{Pro(R|MGT)Pro(MGT)Pro(GT|MGT, R)}{Pro(GT)Pro(MGT|GT, R)} \\ &= Pro(R|MGT) \cdot \frac{Pro(MGT|GT)}{Pro(GT|MGT)} \cdot \frac{Pro(GT|MGT, R)}{Pro(MGT|GT, R)} \\ &= \Phi \cdot Pro(R|MGT). \end{aligned}$$

Both GT and MGT are independent of R, thus, Φ relies on the ratio of $Pro(MGT|GT)/Pro(GT|MGT)$ which is the probability of choosing correct points by the human interaction and can be assumed as a constant by the averaging approach. Therefore, we use the evaluation of $Pro(MGT|R)$ and $Pro(R|MGT)$ to approximate $Pro(GT|R)$ and $Pro(R|GT)$ as illustrated in Fig. 9.

Suppose that the segmentation result is denoted by $R = \{r_1, r_2, \ldots, r_{mi}\}$ and the ground-truth is $MGT = \{mgt_1, \ldots, mgt_{mj}\}$. Each r_i or mgt_j means the point set of a segment. There are mi segments in R and mj segments in MGT. For the evaluation of the proposed segmentation, we adjust the completeness $Pro(MGT|R)$ and correctness in [41], [42], [43] $Pro(R|MGT)$ as

$$\begin{aligned} Pro(MGT|R) &= \frac{1}{mi}\sum_{i=1}^{mi}\left(\frac{\max_{j=1}^{mj}|mgt_j \bigcap r_i|}{|r_i|}\right), \\ Pro(R|MGT) &= \frac{1}{mj}\sum_{j=1}^{mj}\left(\frac{\max_{i=1}^{mi}|r_i \bigcap mgt_j|}{|mgt_j|}\right), \end{aligned} \tag{15}$$

where '$||$' means the cardinality of a set.

$Pro(MGT|R)$ is to measure the ratio between the correctly segmented points and the total points in the result. $Pro(R|MGT)$ is to measure the ratio between the correctly segmented points and the total points in the ground-truth. Both the criterion $Pro(MGT|R)$ and $Pro(R|MGT)$ range from 0 to 1. The problem of Eq. (15) is that if $mj = 1$, $Pro(MGT|R) \equiv 1$, and if $mi = 1$, $Pro(R|MGT) \equiv 1$. Therefore, we choose the minimum of $Pro(MGT|R)$ and $Pro(R|MGT)$ to measure the difference of points between the MGT and R as

$$n_{diff} = \min(Pro(MGT|R), Pro(R|MGT)). \tag{16}$$

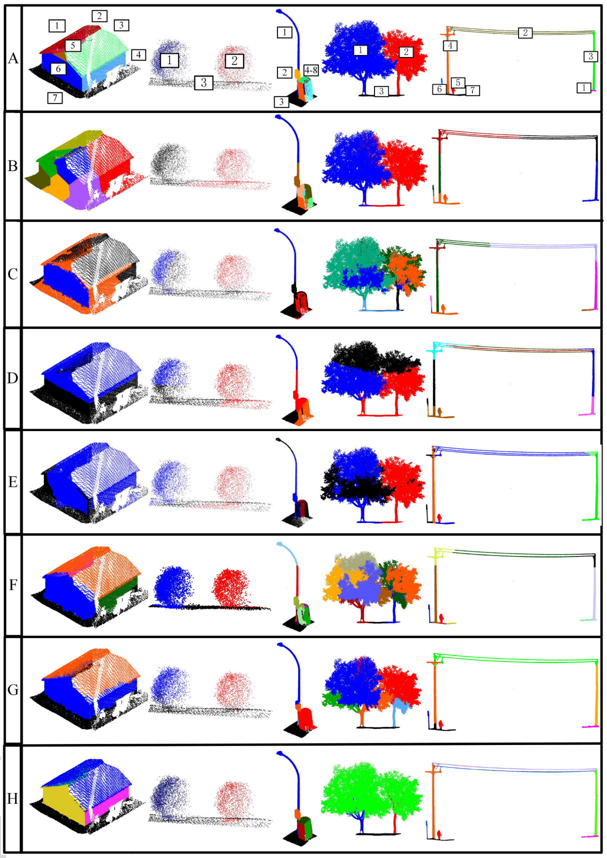


Fig. 10. Performance of different segmentation methods. A.Groundtruth. B.KMiPC. C.KNNiPC. D.3DNCut. E.MinCut. F.PEAC. G.OHC. H. The proposed algorithm.

To balance $Pro(MGT|R)$ and $Pro(R|MGT)$, we choose the criterion F_1-score for evaluating those segments with an imbalanced number of points [44] as

$$n_{F1} = \frac{2 \times (Pro(MGT|R) \cdot Pro(R|MGT))}{Pro(MGT|R) + Pro(R|MGT)}. \tag{17}$$

F_1-score is calculated by the integration of $Pro(MGT|R)$ and $Pro(R|MGT)$ and ranges from 0 to 1.

In the following, we will show the superiority of the proposed algorithm by comparing with region growing-based methods: KMiPC [17] and KNNiPC [45], graph-based methods: 3DNCut [7] and MinCut [28], clustering-based methods: PEAC [18] and OHC [46]. The experimental scenes for testing are shown in Fig. 10, including the HouseSet (7 labels), BushesSet (3 labels), LamppostSet (8 labels), TreesSet (3 labels) and the PowerlinesSet (7 labels). Fig. 10 A shows the MGT of each scene. From B to H are the performance of KMiPC, KNNiPC, 3DNCut, MinCut, PEAC, OHC, and the proposed method, respectively. In the visualization, we use different colors to distinguish segments. In the implementation of the compared methods, KMiPC, KNNiPC, and MinCut are from PointCloudLibrary (www.pointclouds.org/), 3DNCut is extended from the normalized-cut (www.cis.upenn.edu/~jshi/software/) and PEAC is achieved based on the software of [18] (www.merl.com/research/). The graph cut optimization in our work is based on the GCoptimization Library [38], [39], [40], [47]. A brief description of comparison results on each dataset is shown below.

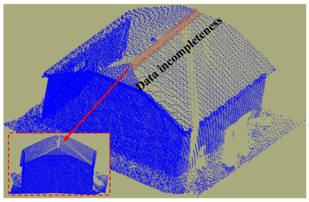


Fig. 11. The incompleteness of the data collection.

In KMiPC and KNNiPC, initial seeds are selected by the euclidean distance information and the growing process is based on the minimization of distances between points and seeds. In 3DNcut and MinCut, the background and foreground are iteratively segmented from scenes. In PEAC and OHC, regions are clustered based on the surface normals and distances between points. In HouseSet, LamppostSet and PowerlinesSet, algorithms KMiPC, KNNiPC, 3DNcut, and MinCut fail to split the connectivity regions, which is their accuracy bottleneck in the segmentation. We fail to separate the two planes for the roof of the house. This is because we lose the segmentation cue in the optimal-vector-field due to the data incompleteness as shown in Fig. 11. In BushesSet, KMiPC, KNNiPC, 3DNcut, and MinCut fail to split the bush and the ground. This shows that they are easy to be affected by the point density. In those three datasets, PEAC, OHC, and the proposed method show a good performance in the separation of connected surfaces. The superiority of the proposed method will be shown in the subsequent numerical value evaluation. In TreeSet, only KMiPC splits the input data into two trees accurately. The segmentation of trees is one of the most difficult tasks in point clouds. Since tree leaves do not form a uniform surface and their divergence will be large, and presumably will be extracted as part of the connectivity region, the proposed algorithm segments tree leaves into very small regions. Therefore, we add a rule in the clustering of points from the non-connectivity region called the small-region-combination (SRC). The corresponding optimal-vector-field for each scene is shown in Fig. 12. As shown in Figs. 12a, 12c, and 12e, the plane intersections have a large optimal-vector-filed at the magnitude, which is quite different from the non-connectivity region. In Figs. 12b and 12d, tree leaves are segmented into pieces and will be combined based on the SRC rule. The following will give a brief discussion about the proposed SRC.

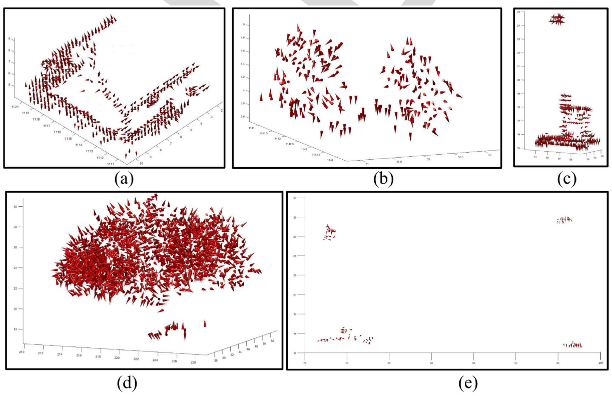


Fig. 12. Optimal-vector-field of each point cloud set. (a) HouseSet. (b) BushesSet. (c) LamppostSet. (d) TreesSet. (e) PowerlinesSet.

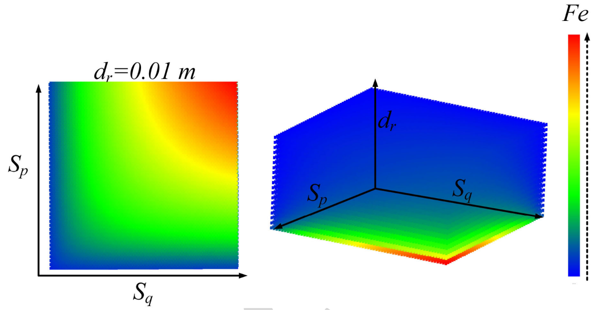


Fig. 13. The change tendency of the energy function. (a) 2D figure of the function when d_r is small. (b) 3D figure of the function.

The segmentation of tree crowns based on the coordinate information is difficult, and it is unavoidable to split tree leaves into pieces. Therefore, we add the SRC algorithm to combine spatially neighboring regions that have large connectivity scores. In order to decide whether to combine regions or not, we define an energy function Fe based on the average connectivity scores and the distance of two regions as

$$Fe = \frac{S_p \cdot S_q}{S_p + S_q} \cdot \frac{1}{d_r}, \tag{18}$$

where S_p and S_q are average connectivity score of two regions, respectively, and d_r is the closest euclidean distance between those two regions. The change tendency of Fe is demonstrated in Fig. 13. Fig. 13a shows that if two regions are very close (i.e., 0.01 m), Fe grows quickly when their connectivity scores are high. Fig. 13b shows that when d_r is increasing, Fe grows slowly even though we enlarge the connectivity scores of regions.

Based on the defined energy function, we set a threshold for cutting off the combination in the SRC. If Fe of two regions is larger than the threshold, those two regions will be combined. When the cut-off threshold is decreasing, more and more regions will be combined. The threshold setting is based on the users' demand of the combination, e.g., treetop leaves, main branch leaves, or all leaves. In the case of trees from Fig. 10, we show results of the SRC when different thresholds are chosen as shown in Fig. 14.

First, we set the cut-off threshold as large as 10.0, and trees are over-segmented into 141 regions as shown in Fig. 14a. Then, we reduce the threshold to 5.0, and segmentation results consist of 86 regions as shown in Fig. 14a. Small regions of tree leaves are combined. Next, we continue to reduce the cut-off threshold to 2.0 and 1.0 as shown in Figs. 14c and 14d, respectively. At this time, the over-segmentation has been improved considerably. Most of the treetop leaves are grouped together. The second row of Figs. 14c and 14d shows the merging of the main branches. Finally, we set the threshold to 0.5 and achieve the segmentation results as shown in Fig. 10.

The numerical value of the evaluation is shown in Table 2. The average accuracy of the experimental scenes shows that our method is more accurate than all the compared methods in terms of the Precision, Recall, n_diff and n_{F1}. The evaluation shows that the bottleneck of the point cloud segmentation, i.e., the split of overlapping regions, can be addressed well by the proposed algorithm.

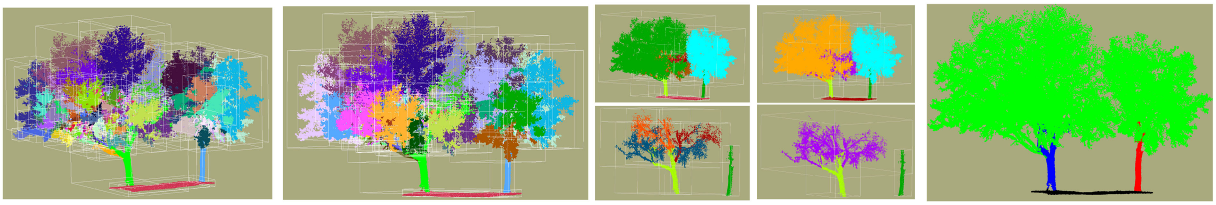


Fig. 14. Segmentation results with SRC when different cut-off thresholds are chosen. (a) Cut-off threshold is 10.0. (b) Cut-off threshold is 5.0. (c) Cut-off threshold is 2.0. (d) Cut-off threshold is 1.0. (e) Cut-off threshold is 0.5.

6.2 Sensitivity Analysis of Parameters

In the algorithm implementation, there are four parameters, namely the k to select the nearest neighbor points, the coefficient λ to balance the terms in the optimal-vector-field calculation, the coefficient γ to balance the data term and smoothness term in the graph-cut segmentation and ϵ used in the data term calculation. In our work, k is 20, λ is 0.9, γ is 0.1, and ϵ is 0.1. For the purpose of the sensitivity analysis, we range all parameters from -30 percent to +30 percent with respect to the suggested values. The analysis is conducted by floating one parameter and fixing the rest of the parameters. The accuracy of the above-mentioned scenes is shown in Fig. 15 using different parameters.

The optimal accuracy in the experimental scenes is $n_{diff} = 90.81\%$ and $n_{F1} = 92.59\%$, respectively. In each case, the mean accuracy of n_{diff} and n_{F1} is no less than 89 and 90 percent, respectively.

In tuning parameters, k and λ are used for the optimal-vector-field procedure. k is usually a necessary parameter in the point cloud processing and is set empirically based on the point density. If the point density falls in 500 to 1000 points/m^2, k is suggested to be between 20 to 30. A larger density scene requires more points to obtain enough neighboring information. A smaller density scene requires fewer points to keep the neighboring information in the local region. A large k may cause the under-segmentation and a small k will increase the over-segmentation rate in results. λ helps obtain the segmentation cue for providing the connectivity information. In the setting of parameters, users choose a proper k based on the density first. Then, tune λ to obtain most connectivity regions from the input scene visually. In our graph-cut procedure, users are required to search γ and ϵ to segment planes. The setting of γ depends on the connectivity region information. If one does not focus on small

TABLE 2
Details of the Evaluation Accuracy

DataSet	Assessment (%)	Methods						
		KMiPC	KNNiPC	3DNCut	MinCut	PEAC	OHC	Proposed
HouseSet	Precision	80.65	81.58	87.98	75.75	94.62	84.10	91.68
	Recall	49.26	71.13	77.15	85.49	68.83	72.23	86.40
	n_{diff}	49.26	71.13	77.15	75.75	68.83	72.23	86.40
	n_{F1}	61.16	76.00	82.21	80.33	79.69	77.71	88.96
BushSet	Precision	76.06	85.26	81.07	90.55	94.54	95.00	97.43
	Recall	84.31	88.94	85.27	87.89	88.86	94.73	93.67
	n_{diff}	76.06	85.26	81.07	87.89	88.86	94.73	93.67
	n_{F1}	79.97	87.06	83.12	89.20	91.61	94.87	95.52
LamppostSet	Precision	80.10	82.33	79.11	65.52	94.36	86.31	90.45
	Recall	78.84	88.96	73.42	78.46	73.85	82.72	90.16
	n_{diff}	78.84	82.33	73.42	65.52	73.85	82.72	90.16
	n_{F1}	79.47	85.52	76.16	71.41	82.85	84.48	90.30
TreeSet	Precision	83.92	79.20	76.06	94.12	94.45	97.37	93.49
	Recall	92.24	61.93	81.47	67.10	51.55	89.71	93.20
	n_{diff}	88.92	61.93	76.06	67.10	51.55	89.71	93.20
	n_{F1}	87.88	69.51	78.67	78.35	66.70	93.38	93.34
PowerLineSet	Precision	64.62	91.61	83.21	74.34	85.50	96.77	99.48
	Recall	81.72	88.98	79.14	96.56	84.93	93.37	90.63
	n_{diff}	64.62	88.98	79.14	74.34	84.93	93.37	90.63
	n_{F1}	72.17	90.28	81.12	84.00	85.21	95.25	94.85
Average	Precision	77.29	84.00	81.49	80.06	91.37	91.91	**94.50**
	Recall	77.69	79.99	79.29	83.10	73.60	86.55	**90.81**
	n_{diff}	71.81	77.93	77.37	74.32	73.60	86.55	**90.81**
	n_{F1}	76.46	81.67	80.26	80.66	81.72	89.14	**92.59**

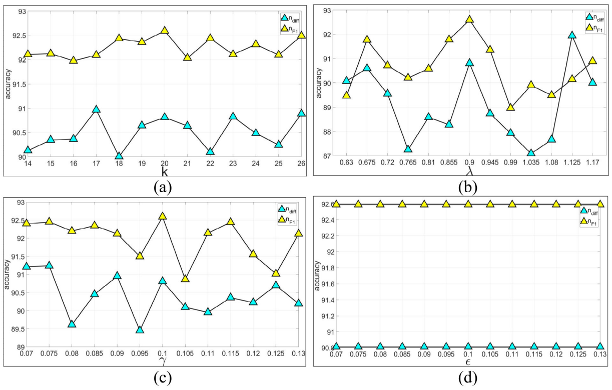


Fig. 15. Sensitivity analysis. (a) Mean of n_{diff} and n_{F1} is 90.49 and 92.24 percent, respectively, when k is floating from -30 to +30 percent. (b) Mean of n_{diff} and n_{F1} is 89.11 and 90.60 percent, respectively, when λ is floating from -30 to +30 percent. (c) Mean of n_{diff} and n_{F1} is 90.40 and 91.97 percent, respectively, when γ is floating from -30 to +30 percent. (d) Mean of n_{diff} and n_{F1} is 90.81 and 92.59 percent, respectively, when ϵ is floating from -30 to +30 percent.

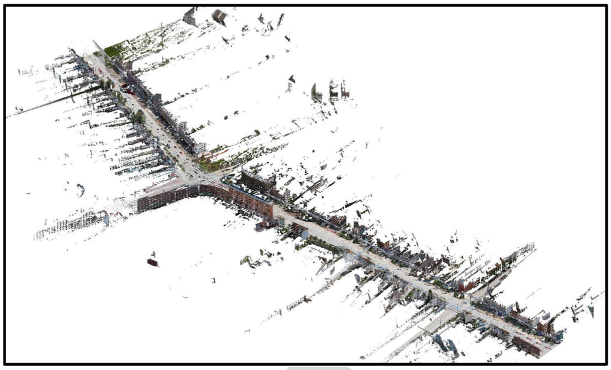


Fig. 16. The input street scene with MLS data.

pieces of planes in a large-scale scene, we suggest increasing γ, which will make the connectivity region smoother. The evaluation shows that the algorithm is quite stable when ϵ is small. Fig. 15 demonstrates that the output of our model can be stable to different input parameters in the suggested range. The spacing between points in the optimal-vector-field formulation is initialized as $\Delta x = 1$ m, $\Delta y = 1$ m and $\Delta z = 1$ m and the time step is set as $\Delta t = 0.18$ based on the convergence rule.

The setting of parameters is procedure-by-procedure. Although a combination of variable changes can be better, only a proper λ will provide a better segmentation cue. In the point cloud processing, the input scene consists of objects different in size. Users may need to obtain small planes from indoor scenes for a fine segmentation or to achieve large building planes from outdoor scenes for 3D modeling, which depends on users' demand. Therefore, we obtain the optimal λ and γ by the grid searching.

Experiments were done on a Windows 10 Enterprise 64-bit, Intel Core i7-6900k, 3.20 GHz processor with 64 GB of RAM, and computations were carried on Matlab. The cost time for each algorithm is shown in Table 3. In the last column, the first part is the time cost of the optimal-vector-field and the second part is for the segmentation. In comparison, only KMiPC and KNNiPC perform better than ours. 3DNCut is slower than the proposed method when the scale of the scene is large. For MinCut, the human-computer interaction is very time-consuming. The organization of point clouds in PEAC is not counted in Table 3, which will cost lots of time. OHC is faster than ours because they choose a sampling strategy to reduce the time cost, which will decrease the segmentation accuracy.

6.3 Performance on Different LiDAR Point Cloud Sets

This section shows our performance on different LiDAR point clouds. First, we will demonstrate the proposed plane segmentation from MLS data. The experiment data are point clouds of a common street scene as shown in Fig. 16. There are 12,181,900 points in this scene and the road is about 1 km long. The challenges in this experiment include (1) the incompleteness of points, (2) the presence of noise and occlusions, and (3) the large volume of points. The segmentation is performed in the area which is 30 meters to the center of the road, and our results are shown in Fig. 17. As shown in the given four close-views, different planes are visualized in different colors.

Two points are worth noting in MLS data:

1) The outliers removal is achieved by computing the mean m and standard deviation δ of k-nearest neighbor distance. Points that fall outside $m \pm \delta$ will be regarded as outliers as shown in Fig. 18a and removed before the segmentation.
2) In MLS data, the density of ground points is much larger than off-ground points. Therefore, we extract ground to reduce the volume of input points using the elevation difference [48] as shown in Fig. 18b.

Second, we demonstrate the plane segmentation on ALS data. The experiment dataset is from Dublin project (doi: 10.17609 / N8MQ0N). This dataset is collected by aerial

TABLE 3
Execution Time of Each Algorithm

Dataset	Number of points	Density (points/m^2)	Time cost (seconds)						
			KMiPC	KNNiPC	3DNCut	MinCut	PEAC	OHC	Proposed
HouseSet	73899	821	0.72	0.76	61.67	15.99	4.92	10.63	6.65+10.45
BushesSet	7046	793	0.04	0.04	0.60	3.16	0.24	1.63	0.40+1.61
LamppostSet	52403	4645	0.46	0.21	29.96	31.92	3.02	9.91	2.10+9.82
TreesSet	257469	858	1.86	1.84	520.14	122.21	111.32	35.96	16.15+60.47
PowerlinesSet	154307	1836	1.07	0.58	234.35	139.53	33.65	16.23	9.75+35.07

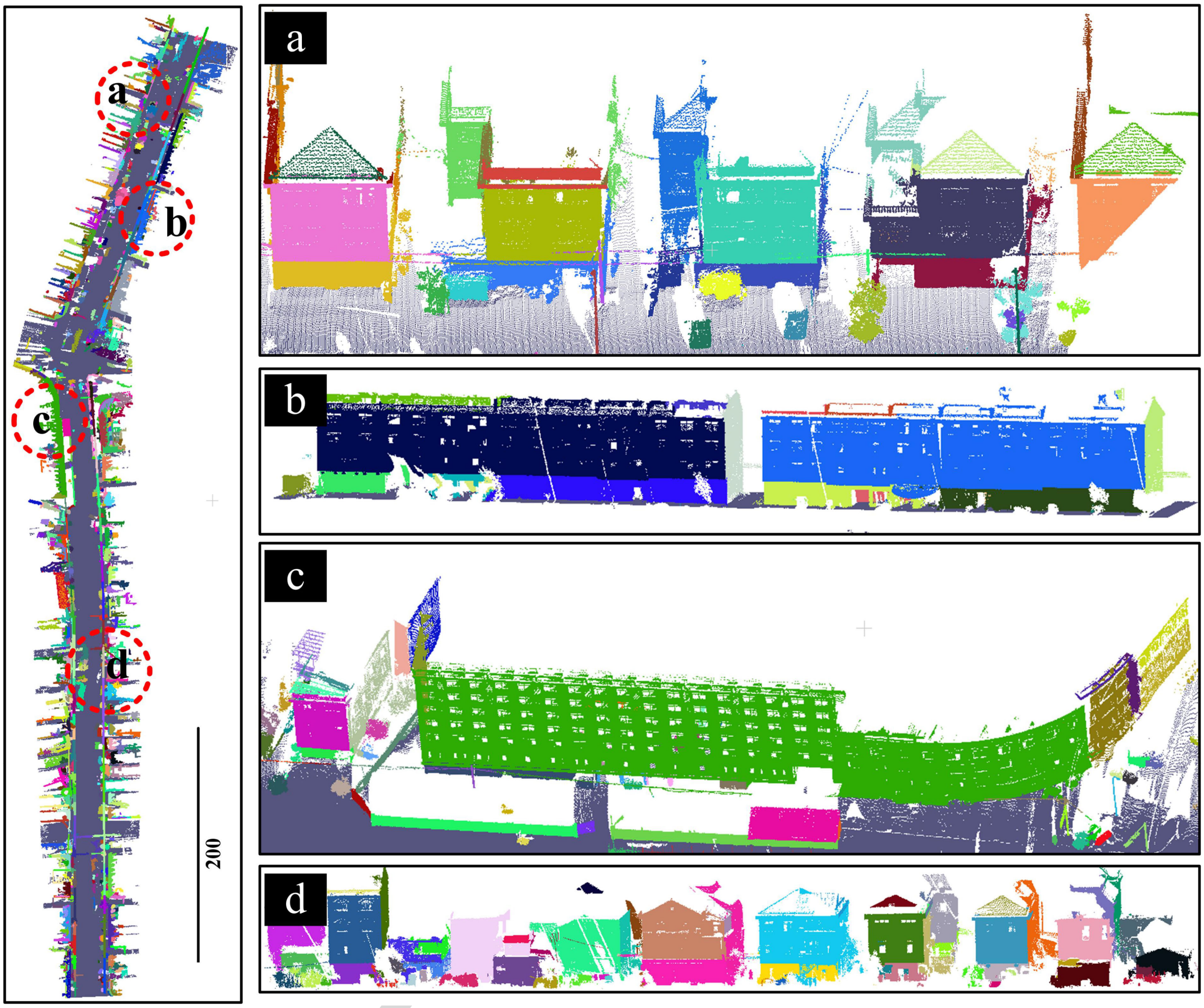


Fig. 17. Plane segmentation from MLS data.

laser scanning (ALS) in the form of a 3D point-cloud (LAZ) and imagery data. Data were obtained at an average flying altitude of 300 meters. The main challenge in this experiment is the incompleteness of the object information. In ALS data, the facades of buildings are missing due to the fact that the laser beam is scanned from top to bottom. As shown in Fig. 19, the first row shows 2D images of the experimental scenes and the second row shows the corresponding 3D-point data. Fig. 20 shows our segmentation results. Fig. 19a shows that sedans are usually in only one plane. Treetop leaves are grouped into one unit as shown in Figs. 20a and 20b based on the proposed SRC algorithm. Fig. 20c shows that the proposed method is adaptive to different geometric surfaces. From the formulation of the optimal-vector-field, all building tops and the ground have the same surface normal. If the building tops are in different elevations, we can split them easily as shown in the regions #1 to #5 in Fig. 21. However, since the ALS is scanned from top-down, the magnitude of the optimal-vector-field is consistently low in almost all places, surface normals have the

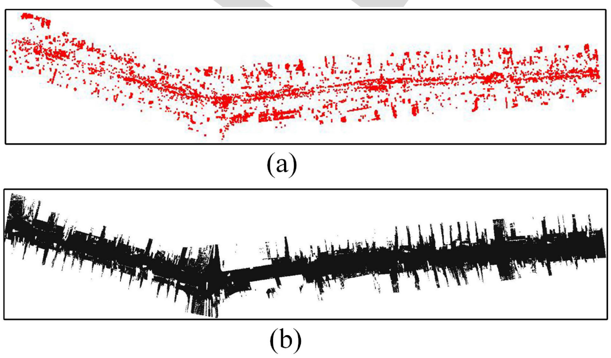


Fig. 18. Illustration of points that are worth noting in MLS data. (a) Outliers removal. (b) Extracted ground points.

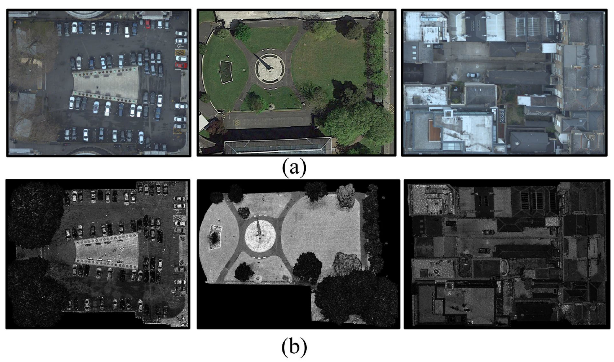
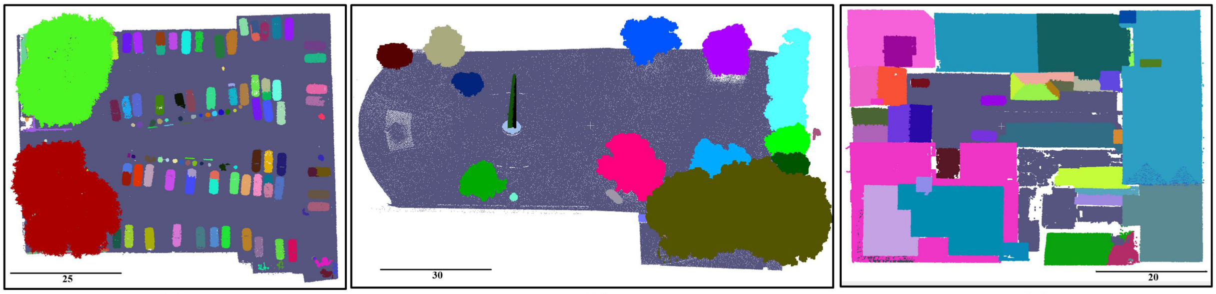


Fig. 19. Input scenes with ALS data. (a) 2D imageries of input scenes. (b) 3D ALS point clouds of input scenes.



(a) (b) (c)

Fig. 20. Plane segmentation from ALS data. (a) Parking lot scene. (b) Park scene. (c) Street block scene.

same direction at building-to-building and ground-to-building, it is difficult to split connected planes as shown in the regions #a to #d in Fig. 21. Although there are four different planes in the region #1 in Fig. 21, the proposed plane segmentation regards them as one large plane.

6.4 Application to the Multi-Object Segmentation and Detection

In order to show the benefit of the proposed algorithm to the individual object segmentation, this subsection shows how to merge planes into complete objects based on the compatibility of information, including the intensity and color. In the object segmentation, we do not have any prior knowledge of the input scene, e.g., the number of objects or the recognition of objects. Therefore, we have to analyze the compatibility of neighboring information, e.g., property and structure. The compatibility of the property (e.g., the material, color, and texture), which results in the discontinuity of the gray value, is commonly used in the 2D image segmentation. The compatibility of the structure is mostly used for the segmentation of objects with a specific shape, such as the hedgehog shape segmentation [49] and the thin structure estimation [50]. Because the topology determination from point clouds is still a challenging task [51], we did not use the structure compatibility for the segmentation. In the merging of planes, two regions will be combined if

$$|\overline{\phi_A} - \overline{\phi_B}| < T, \tag{19}$$

where $\overline{\phi_A}$ and $\overline{\phi_B}$ are the mean value of a plane A and B, respectively, using specific compatibility information. T is the user-defined threshold for the grouping. The following section shows the performance of merging planes into complete objects using different compatibility information.

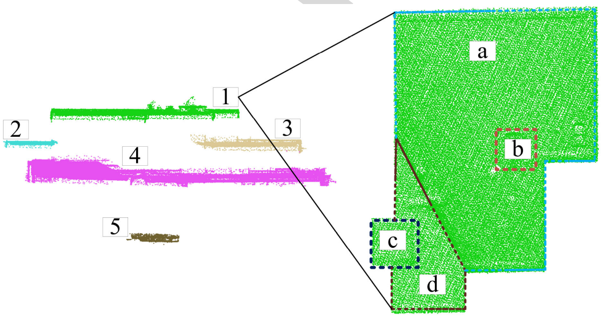


Fig. 21. Plan segmentation results of roofs at different elevations.

First is the merging of planes from MLS data using the color compatibility. The color information (RGB) is obtained by the registration of LiDAR point clouds and images. In our work, we transform RGB into HSV (Hue, Saturation, Value) space [52]. Hue is defined as an angle in the range from 0 to 2Π. The threshold T used in the merging is $10°$. The merging result is shown in Fig. 22. Each bounding box in Fig. 22 means the combination result of an individual object. Planes from a building are merged into one individual object. We obtain 74/74 buildings from the test street scene using the compatibility of color information, which is a promising result in the segmentation of MLS data. A close-view of a common traffic scene in the segmentation is shown in Fig. 23. We segment eight individual objects accurately, including two vehicles, two groups of vegetation, two human beings, one traffic light, and one building. The problem of the merging based on the color information is that since the color is assigned based on the registration between images and point clouds, the color can be unreliable in the connectivity region.

Second is the merging of planes using the intensity compatibility. The intensity information contains only one channel and is scaled to [0,255]. In the merging process, T is set as 20. Our results segment multiple vehicles in a parking lot as shown in Fig. 24a and obtain different roofs from a block scene as shown in Fig. 24c. Errors appear in the segmentation of trees as shown in Fig. 24b, i.e., spatially close trees are grouped together. The problem of the merging based on the intensity information is that the intensity highly depends on the collection system and has to be calibrated thoroughly before the merging [53].

For the comparison, we choose the well-known dataset Semantic3D [54], which is the largest available LiDAR dataset with over billion points from a variety of urban and rural scenes. Each point has RGB and intensity values (the latter of which we do not use). There are eight classes in the benchmark, namely man-made ground: mostly pavement and road, natural ground: mostly grass, high vegetation: trees and large bushes, low vegetation: flowers or small bushes which are smaller than 2 m, buildings: tenements and facades, hard scape: a class with for instance garden walls and fences, scanning artifacts: artifacts caused by dynamically moving objects and cars. The comparison contains SnapNet_[55], SEGCloud [56], SPGraph [57], shellnet_v2[58], RGNet [59], KP-FCNN [60], OctreeNet_CRF [61], GAC [62] and ours. Performance is shown in Fig. 25 and evaluated based on the per-class accuracy (Acc) and average accuracy, which is defined as the proportion of

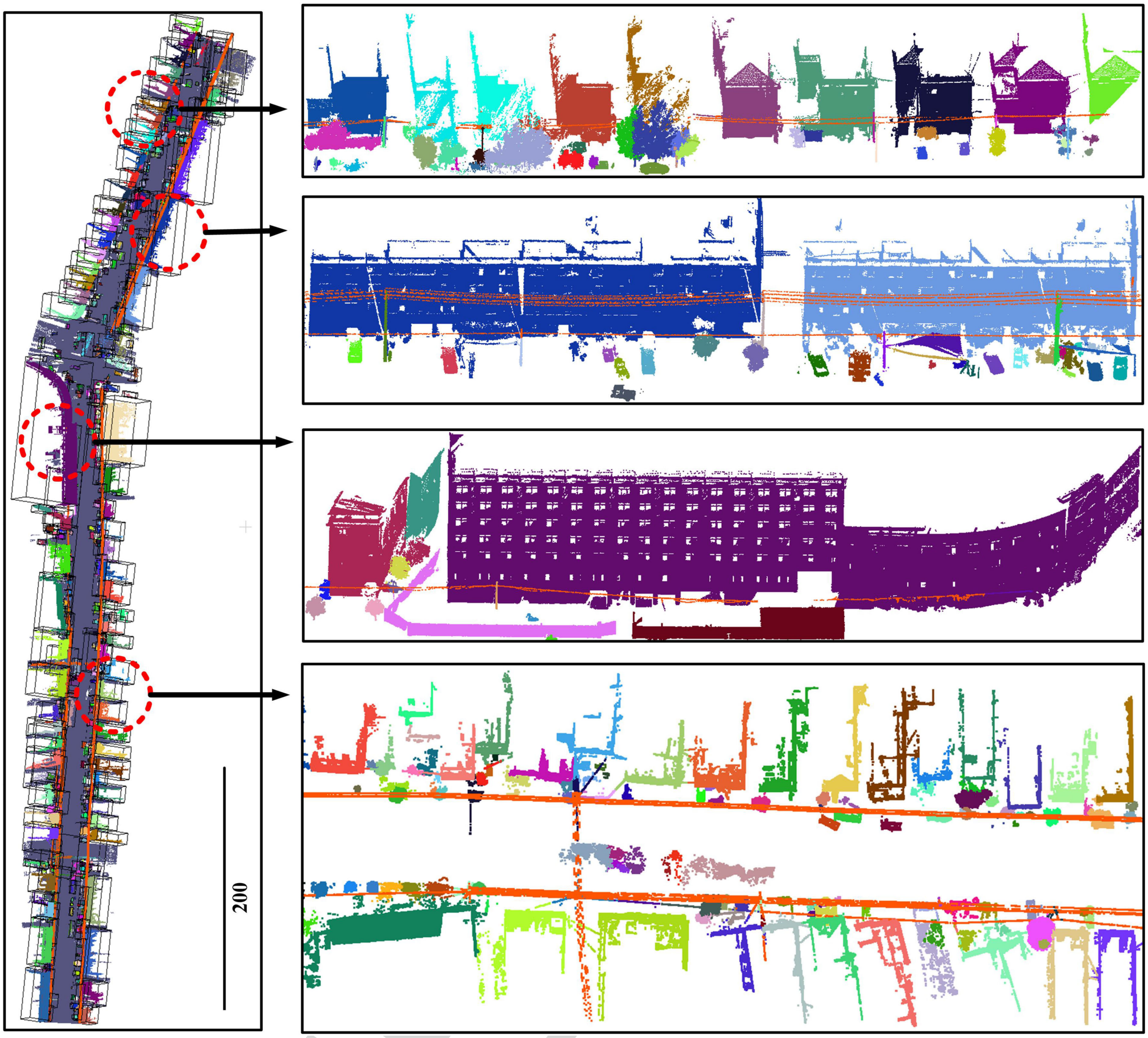


Fig. 22. Merging of planes from MLS data.

correctly segmented points, as shown in Table 4. Their accuracy is based on the authors' published work. The classification of our segments is by setting thresholds on the volume, elevation, and normal vectors at points.

Although our unsupervised classification step does not require the training process, the accuracy highly depends on the prior knowledge of objects, such as the volume, elevation, and normal vectors at points. The procedure of the classification is from bottom to top. First, the distinguishment of the ground and non-ground regions is based on elevation information. Second, non-ground objects consist of planes located around the ground points' boundary are recognized as buildings. Planes are indicated by the normal vector information at points. Third, the classification of the hard scape, cars, or vegetation regions from above-ground points is solved by using the template matching approach. We formulate templates for the vehicle and vegetation by different cubes and poles respectively and try to match the template with the non-ground points to classify cars and trees. The matching process requires users to keep adding samples into templates to obtain a threshold range for each class. This is because objects are often incomplete due to occlusion in the data collection. The classification is implemented using a decision tree strategy based on the achieved thresholds automatically. The rest of the non-ground objects are regarded as points from the hard scape.

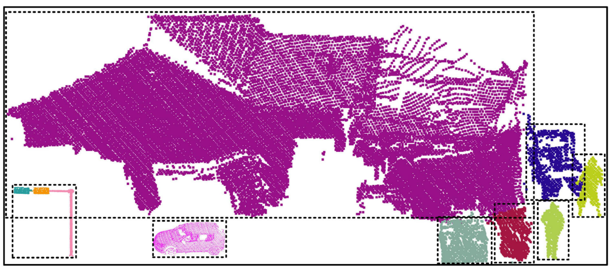


Fig. 23. Details of merged objects from MLS data.

Our misclassification points are from the overlapping regions between objects and ground points. As shown in Table 4, the accuracy of our method is higher than the above-mentioned methods if most of the components from the input scene have an accurate vector-field, e.g., cars and plane regions. Results show that the proposed method

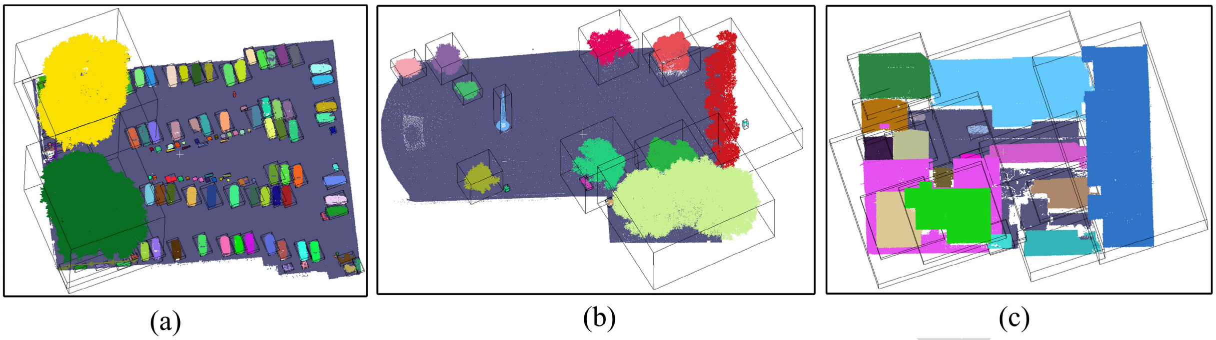


Fig. 24. Merging of planes from ALS data. (a) Parking lot scene. (b) Park scene. (c) Street block scene.

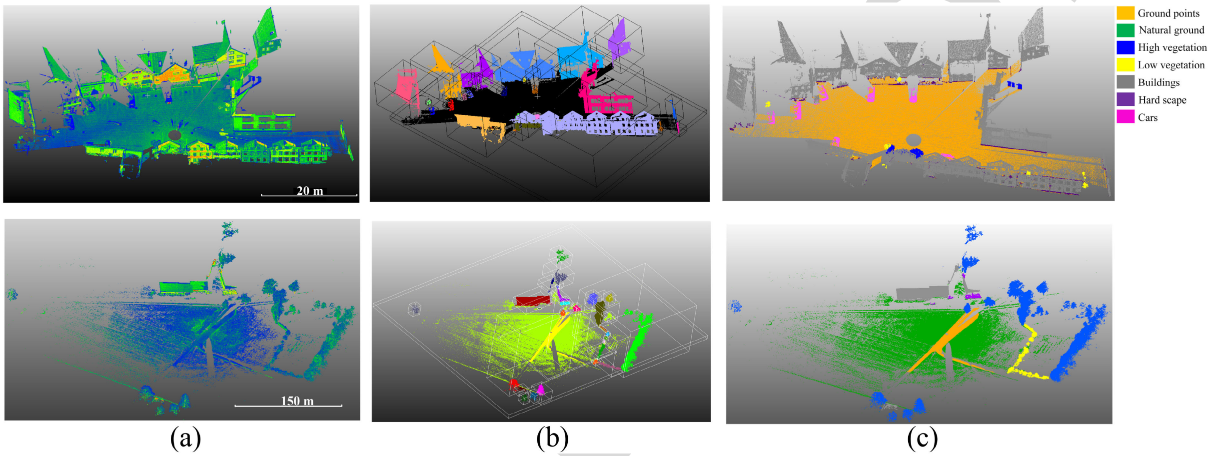


Fig. 25. Detection of different objects. (a) Input point clouds. (b) Segmentation results. (c) Detection results.

TABLE 4
Comparison With the Object Segmentation Algorithms

Method	Average	man-made ground	natural ground	high vegetation	low vegetation	buildings	hard scape	scanning artifacts	cars
SnapNet_	0.591	0.820	0.773	0.797	0.229	0.911	0.184	0.373	0.644
SEGCloud	0.613	0.839	0.660	0.860	0.405	0.911	0.309	0.275	0.643
SPGraph	0.732	0.974	0.926	0.879	0.440	0.932	0.310	0.635	0.762
shellnet_v2	0.693	0.963	0.904	0.839	0.410	0.942	0.347	0.439	0.702
RGNet	0.747	0.975	0.930	0.881	0.481	0.946	0.362	0.720	0.680
KP-FCNN	0.746	0.909	0.822	0.842	0.479	**0.949**	**0.400**	**0.773**	0.797
OctreeNet_CRF	0.591	0.907	0.820	0.824	0.393	0.900	0.109	0.312	0.460
GAC	0.708	0.864	0.777	**0.885**	**0.606**	0.942	0.373	0.435	0.778
Ours	**0.754**	**0.985**	**0.965**	0.821	0.442	0.905	0.245	N/A	**0.914**

performs better on smooth surfaces and achieves the highest average accuracy, this is because the vector-field of smooth surfaces is calculated well in the detection of the intersection between different planes than vegetation regions.

6.5 Advantages and Limitations

The above experiments show that our method achieves the plane segmentation accurately and succeed in detecting the overlapping intersections. Compared with the work which applied graph-cut directly based on the color consistency, e.g., [27], our advantage is that we do not require the color information in the segmentation. Our segmentation depends on the coordinate only. The color in point clouds is assigned based on the registration between images and point clouds, which is a not well-addressed task and can be unreliable in connectivity regions. Compared with the work which applied graph-cut to point clouds based on the foreground and background separation, e.g., [28], we do not need human-computer interaction. Compared with the work which applied normalize-cut to the point cloud segmentation, e.g., [7], we achieve the segmentation using two labels only, therefore, we do not need to initialize the number of targets. Compared with the work which obtains planes based on the merging of points or supervoxels, e.g., [46], we do not have the iterative merging process and succeed in ensuring the optimization by a developed binary segmentation model. Since the graph-cut is weak in the segmentation of thin structures, which may cause problems in the connectivity region segmentation, one may have to decrease the λ to make the connectivity region thick.

Compared with deep learning methods, we obtain a better performance on the scene of cars and ground points, because of their accurate optimal-vector-field. The drawbacks lie in the detection of vegetation, due to the fact that the vector-field there is massive and scatter. We do not require a training processing in the segmentation, but the threshold setting is necessary for the purpose of the classification.

The similar work of using the vector field for the geometry analysis has been used in [63]. We clarify the difference in this subsection. They identify parts of the shape by defining deformation energy on the shape and find a decomposition of the shape. However, they do not show how to split different surfaces and planes. In this paper, the vector field is optimized to cue the intersection regions of planes, which is new and effective to the plane segmentation. However, (1) since the segmentation is in the primitive-level, we are required to add the merging processing to obtain complete objects; (2) the algorithm asks for a line fitting process to deal with the linear objects. e.g., power lines and curb edges; our accuracy is degraded in the segmentation of trees, due to the non-uniform surface there; the proposed method is not suitable for the fine segmentation, because the optimal-vector-field of small planes can be affected by outliers easily; (3) points from the intersection regions are missing in the segmentation results, which degrades the completeness accuracy from the proposed algorithm.

7 Conclusion

This paper proposes a new strategy of the plane segmentation for LiDAR point clouds. The algorithm mainly has two phases to add both the local and global constraints for the segmentation. First, a new optimal-vector-field is formulated to detect the plane intersections in a local phase. Second, the input scene is divided into the connectivity and non-connectivity region by a single graph-cut model in a global phase. Segmentation cues are inferred by the formulated optimal-vector-field effectively and used for the plane segmentation. Experiments show that the proposed segmentation works accurately on both mobile and airborne LiDAR point clouds with an average precision of 94.50 percent and the recall of 90.81 percent. The achieved plane segmentation results can be easily merged into complete objects based on the color and intensity information, which are better than state-of-the-art supervised learning methods with an average accuracy of 75.4 percent.

It could be expected that improving plane intersection detection will result in increasing the accuracy of plane segmentation and individual object detection. Besides, 3D urban scene understanding, e.g., 3D object detection and classification, will increasingly rely on 3D laser scanning data, hence, further considerable research is required to address the issue of merging complex components, e.g., incomplete or sparse objects.

Acknowledgments

This work was supported by the Natural Science Foundation of the Higher Education Institutions of Jiangsu Province (19KJB520010), the National Key Research and Development Program of China (2019YFD1100404), China Postdoctoral Science Foundation (2019M661852), the Natural Science Foundation of Jiangsu Province (BK20171453), and the Central Public-interest Scientific Institution Basal Research Fund (CAFYBB2019QD003). This work was also supported by the National Sciences and Engineering Research Council (NSERC) in Canada and the Alberta Innovates Technology Futures (AITF) Scholarships.

References

[1] D. Chen, L. Zhang, P. T. Mathiopoulos, and X. Huang, "A methodology for automated segmentation and reconstruction of urban 3-D buildings from ALS point clouds," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 7, no. 10, pp. 4199–4217, Oct. 2014.

[2] T. Yun, F. An, W. Li, Y. Sun, L. Cao, and L. Xue, "A novel approach for retrieving tree leaf area from ground-based liDAR," *Remote Sens.*, vol. 8, no. 11, 2016, Art. no. 942.

[3] H. Zheng, R. Wang, and S. Xu, "Recognizing street lighting poles from mobile liDAR data," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 1, pp. 407–420, Jan. 2017.

[4] S. Xia and R. Wang, "A fast edge extraction method for mobile liDAR point clouds," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 8, pp. 1288–1292, Aug. 2017.

[5] L. Nan and P. Wonka, "PolyFit: Polygonal surface reconstruction from point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2353–2361.

[6] R. Qiu, Q.-Y. Zhou, and U. Neumann, "Pipe-run extraction and reconstruction from point clouds," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 17–30.

[7] Y. Yu, J. Li, H. Guan, C. Wang, and J. Yu, "Semiautomated extraction of street light poles from mobile liDAR point-clouds," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 3, pp. 1374–1386, Mar. 2015.

[8] A. Golovinskiy, V. G. Kim, and T. Funkhouser, "Shape-based recognition of 3D point clouds in urban environments," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, 2009, pp. 2154–2161.

[9] A. Dai and M. Nießner, "3DMV: Joint 3D-multi-view prediction for 3D semantic scene segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 452–468.

[10] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3D object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4490–4499.

[11] A. Kaiser, J. A. Ybanez Zepeda, and T. Boubekeur, "A survey of simple geometric primitives detection methods for captured 3D data," in *Computer Graphics Forum*, vol. 38. Hoboken, NJ, USA: Wiley, 2019, pp. 167–196.

[12] D. Tóvári and N. Pfeifer, "Segmentation based robust interpolation-a new approach to laser data filtering," *Int. Archives Photogrammetry Remote Sens. Spatial Inf. Sci.*, vol. 36, no. 3/19, pp. 79–84, 2005.

[13] 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.

[14] R. Schnabel, R. Wahl, and R. Klein, "Efficient RANSAC for point-cloud shape detection," in *Computer Graphics Forum*, vol. 26. Hoboken, NJ, USA: Wiley, 2007, pp. 214–226.

[15] H. Lin *et al.*, "Semantic decomposition and reconstruction of residential scenes from liDAR data," *ACM Trans. Graph.*, vol. 32, no. 4, 2013, Art. no. 66.

[16] G. Lavoué, F. Dupont, and A. Baskurt, "A new cad mesh segmentation method, based on curvature tensor analysis," *Comput.-Aided Des.*, vol. 37, no. 10, pp. 975–987, 2005.

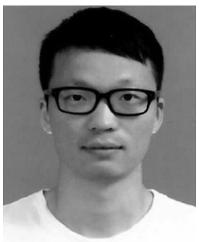
[17] R. A. Kuçak, E. Özdemir, and S. Erol, "The segmentation of point clouds with k-means and artifical neural network," *Int. Archives Photogrammetry Remote Sens. Spatial Inf. Sci.*, vol. XLII-1/W1, pp. 595–598, 2017.

[18] C. Feng, Y. Taguchi, and V. R. Kamat, "Fast plane extraction in organized point clouds using agglomerative hierarchical clustering," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 6218–6225.

[19] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

[20] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, Nov. 2001.

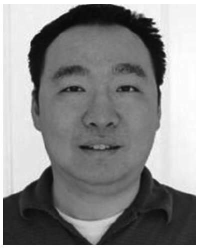
[21] W. Tao, H. Jin, and Y. Zhang, "Color image segmentation based on mean shift and normalized cuts," *IEEE Trans. Syst., Man., Cybern., B, Cybern.*, vol. 37, no. 5, pp. 1382–1389, Oct. 2007.
[22] J. Malik, S. Belongie, T. Leung, and J. Shi, "Contour and texture analysis for image segmentation," *Int. J. Comput. Vis.*, vol. 43, no. 1, pp. 7–27, 2001.
[23] B. Yang and Z. Dong, "A shape-based segmentation method for mobile laser scanning point clouds," *ISPRS J. Photogrammetry Remote Sens.*, vol. 81, pp. 19–30, 2013.
[24] P. Polewski, W. Yao, M. Heurich, P. Krzystek, and U. Stilla, "Detection of fallen trees in als point clouds using a normalized cut approach trained by simulation," *ISPRS J. Photogrammetry Remote Sens.*, vol. 105, pp. 252–271, 2015.
[25] Y. Boykov and O. Veksler, "Graph cuts in vision and graphics: Theories and applications," *Handbook Math. Models Comput. Vis.*, pp. 79–96, 2006.
[26] L. Gorelick, O. Veksler, Y. Boykov, and C. Nieuwenhuis, "Convexity shape prior for binary segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 2, pp. 258–271, Feb. 2017.
[27] J. Strom, A. Richardson, and E. Olson, "Graph-based segmentation for colored 3D laser point clouds," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2010, pp. 2131–2136.
[28] A. Golovinskiy and T. Funkhouser, "Min-cut based segmentation of point clouds," in *Proc. IEEE 12th Int. Conf. Comput. Vis. Workshops*, 2009, pp. 39–46.
[29] H. Huang, D. Li, H. Zhang, U. Ascher, and D. Cohen-Or, "Consolidation of unorganized point clouds for surface reconstruction," *ACM Trans. Graph.*, vol. 28, 2009, Art. no. 176.
[30] S. Xu, R. Wang, and H. Zheng, "Road curb extraction from mobile liDAR point clouds," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 2, pp. 996–1009, Feb. 2017.
[31] R. Courant and D. Hilbert, *Methods of Mathematical Physics (Chapter 4)*, vol. 1, Cambridge, U. K.: Cambridge Univ. Press, 1965.
[32] R. Courant, K. Friedrichs, and H. Lewy, "On the partial difference equations of mathematical physics," *IBM J.*, vol. 11, no. 2, pp. 215–234, 1967.
[33] P. D. Lax and R. D. Richtmyer, "Survey of the stability of linear finite difference equations," *Commun. Pure Appl. Math.*, vol. 9, no. 2, pp. 267–293, 1956.
[34] J. G. Charney, R. Fjörtoft, and J. V. Neumann, "Numerical integration of the barotropic vorticity equation," *Tellus*, vol. 2, no. 4, pp. 237–254, 1950.
[35] C. Dorai and A. K. Jain, "Cosmos-a representation scheme for 3D free-form objects," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 10, pp. 1115–1130, Oct. 1997.
[36] J. Liang, R. Lai, T. W. Wong, and H. Zhao, "Geometric understanding of point clouds using laplace-beltrami operator," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 214–221.
[37] D. Boscaini, J. Masci, E. Rodolà, M. M. Bronstein, and D. Cremers, "Anisotropic diffusion descriptors," in *Computer Graphics Forum*, vol. 35. Hoboken, NJ, USA: Wiley, 2016, pp. 431–441.
[38] B. Fulkerson, A. Vedaldi, and S. Soatto, "Class segmentation and object localization with superpixel neighborhoods," in *Proc. Int. Conf. Comput. Vis.*, Oct. 2009, pp. 670–677.
[39] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1124–1137, Sep. 2004.
[40] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 2, pp. 147–159, Feb. 2004.
[41] B. Yang, L. Fang, and J. Li, "Semi-automated extraction and delineation of 3D roads of street scene from mobile laser scanning point clouds," *ISPRS J. Photogrammetry Remote Sens.*, vol. 79, pp. 80–93, 2013.
[42] A. Boyko and T. Funkhouser, "Extracting roads from dense point clouds in large scale urban environment," *ISPRS J. Photogrammetry Remote Sens.*, vol. 66, no. 6, pp. S2–S12, 2011.
[43] P. Kumar, C. P. McElhinney, P. Lewis, and T. McCarthy, "An automated algorithm for extracting road edges from terrestrial mobile lidar data," *ISPRS J. Photogrammetry Remote Sens.*, vol. 85, pp. 44–55, 2013.
[44] C. Goutte and E. Gaussier, "A probabilistic interpretation of precision, recall and F-score, with implication for evaluation," in *Proc. Eur. Conf. Inf. Retrieval*, 2005, pp. 345–359.
[45] B. Douillard *et al.*, "On the segmentation of 3D liDAR point clouds," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 2798–2805.
[46] S. Xu, R. Wang, H. Wang, and H. Zheng, "An optimal hierarchical clustering approach to mobile liDAR point clouds," *IEEE Trans. Intell. Transp. Syst.*, early access, May 02, 2019, doi: 10.1109/TITS.2019.2912455.
[47] Y. Boykov, O. Veksler, and R. Zabih, "Efficient approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intelli.*, vol. 20, no. 12, pp. 1222–1239, Nov. 2001.
[48] K. Zhang, S.-C. Chen, D. Whitman, M.-L. Shyu, J. Yan, and C. Zhang, "A progressive morphological filter for removing nonground measurements from airborne liDAR data," *IEEE Trans. Geosci. Remote Sens.*, vol. 41, no. 4, pp. 872–882, Apr. 2003.
[49] H. Isack, O. Veksler, M. Sonka, and Y. Boykov, "Hedgehog shape priors for multi-object segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2434–2442.
[50] D. Marin, Y. Zhong, M. Drangova, and Y. Boykov, "Thin structure estimation with curvature regularization," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 397–405.
[51] J.-D. Favreau, F. Lafarge, and A. Bousseau, "Fidelity versus simplicity: A global approach to line drawing vectorization," *ACM Trans. Graph.*, vol. 35, no. 4, 2016, Art. no. 120.
[52] S. Sural, G. Qian, and S. Pramanik, "Segmentation and histogram generation using the HSV color space for image retrieval," in *Proc. Int. Conf. Image Process.*, 2002, p. II.
[53] W. Fang, X. Huang, F. Zhang, and D. Li, "Intensity correction of terrestrial laser scanning data by estimating laser transmission function," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 2, pp. 942–951, Feb. 2015.
[54] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, and M. Pollefeys, "Semantic3D. net: A new large-scale point cloud classification benchmark," 2017, *arXiv: 1704.03847*.
[55] A. Boulch, B. Le Saux , and N. Audebert, "Unstructured point cloud semantic labeling using deep segmentation networks," in *Proc. Eurographics Workshop 3D Object Retrieval*, 2017, Art. no. 7.
[56] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese, "Segcloud: Semantic segmentation of 3D point clouds," in *Proc. Int. Conf. Vis.*, 2017, pp. 537–547.
[57] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4558–4567.
[58] Z. Zhang, B.-S. Hua, and S.-K. Yeung, "Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 1607–1616.
[59] G. Truong, S. Z. Gilani, S. M. S. Islam, and D. Suter, "Fast point cloud registration using semantic segmentation," in *Proc. Digital Image Comput.: Techn. Appl.*, 2019, pp. 1–8.
[60] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "KPConv: Flexible and deformable convolution for point clouds," 2019, *arXiv: 1904.08889*.
[61] F. Wang, Y. Zhuang, H. Gu, and H. Hu, "Octreenet: A novel sparse 3-D convolutional neural network for real-time 3-D outdoor scene analysis," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 2, pp. 735–747, Apr. 2020.
[62] L. Wang, Y. Huang, Y. Hou, S. Zhang, and J. Shan, "Graph attention convolution for point cloud semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 10 296–10 305.
[63] Q.-X. Huang, M. Wicke, B. Adams, and L. Guibas, "Shape decomposition using modal analysis," in *Computer Graphics Forum*, vol. 28. Hoboken, NJ, USA: Wiley, 2009, pp. 407–416.



Sheng Xu received the BEng degree in computer science and technology from Nanjing Forestry University, in 2010, and the PhD degree in digital image systems from the University of Calgary, in 2018, respectively. In 2018, he joined the College of Information Science and Technology, Nanjing Forestry University, where he is currently an associate professor. His current research interests include mobile mapping, vegetation mapping, and computer vision.

Ruisheng Wang received the BEng degree in photogrammetry and remote sensing from the Wuhan University, the MScE degree in geomatics engineering from the University of New Brunswick, and the PhD degree in electrical and computer engineering from the McGill University. He is currently an associate professor with the Department of Geomatics Engineering, University of Calgary which he joined in 2012. Prior to that, he worked as an industrial researcher at NOKIA in Chicago, USA since 2008. His research interests include geomatics and computer vision.

Hao Wang received the BEng degree from Tongji University, and the PhD degree from Nanjing Forestry University. He is currently a professor with the College of Landscape Architecture at Nanjing Forestry University which he joined in 1983. He is a member of the editorial committee of Chinese Landscape Architecture. His research interests include landscape architecture design, urban planning and green space system analysis.



Ruigang Yang (Senior Member, IEEE) received the MS degree from Columbia University, and the PhD degree from the University of North Carolina, Chapel Hill. He is currently a full professor of Computer Science at the University of Kentucky. His research interests include computer graphics and computer vision, in particular in 3D reconstruction and 3D data analysis. He has published more than 100 papers, which, according to Google Scholar, has received close to 6,000 citations with an h-index of 37 (as of 2014). He has received many awards, including the US NSF Career award in 2004 and the deans research Award at the University of Kentucky in 2013.