MDPI

Article

Segmentation-Based Classification for 3D Point Clouds in the Road Environment

Binbin Xiang[1], Jian Yao[1,*], Xiaohu Lu[1], Li Li[1], Renping Xie[1], and Jie Li[2]

[1]Computer Vision and Remote Sensing (CVRS) Lab, School of Remote Sensing and Information Engineering, Wuhan University, Wuhan, Hubei, P.R. China

[2]School of Electrical and Electronic Engineering, Hubei University of Technology, Wuhan, Hubei, China

* Correspondence: jian.yao@whu.edu.cn; Tel.: +86-27-68771218; Web: http://cvrs.whu.edu.cn/

Version May 5, 2017 submitted to Remote Sens.; Typeset by LaTeX using class file mdpi.cls

Abstract: The 3D point cloud classification in urban scenes has been widely applied in the fields of automatic driving, map updating, change detection, etc. Accurate and effective classification of mobile laser scanning (MLS) point clouds remains a big challenge for these applications. In this paper, we propose a unified framework to classify 3D urban point clouds acquired in the road environment. At first, an efficient 3D point cloud segmentation approach is applied to generate segments for further classification. This is achieved by using the Pairwise Linkage (P-Linkage) algorithm for the initial point clouds segmentation followed by our proposed two-step post-processing approach to improve the original segmentation results for accurate classification. Secondly, a set of novel features are extracted from each segment and an effective classifier for training and testing is used. The good performance of the extracted features is determined by employing three popularly used classifiers, Support Vector Machine (SVM), Random Forests (RF) and Extreme Learning Machine (ELM), respectively. Thirdly, the contextual constraints among objects are used to further refine the classification results based on segments via graph cuts. A set of experiments on our own manually labelled dataset show that our proposed framework can effectively segment the testing point clouds. On the test dataset, the initial classification can reach a high precision of 80.8%−92.9% and a good recall rate of 77.5%−79.2%, respectively. After the classification refinement via graph cuts, the precision and recall rate are increased about 0.3% and 3.1%, respectively. These experimental results convincingly prove that our proposed framework is effective for classifying 3D urban point clouds acquired by a vehicle LiDAR system in the road environment.

Keywords: Point Classification, Graph Cuts, Pairwise Linkage Segmentation, Support Vector Machine, Mobile Laser Scanning

1 Introduction

The accurate 3D spatial information has been attracted considerable interest in recent years due to the increasing demand of the scene understanding and detailed semantic analysis in road environments. As the rapid advancements of 3D laser scanning technology, accurate 3D point clouds of large areas can be obtained easily and cheaply [1]. A common way to quickly collect 3D data of urban road environments is by using the mobile laser scanning (MLS) technology. The 3D information acquired by the MLS technology can be applied to complete various missions. For example, in road environments, the efficient collection of accurate 3D data can benefit future driver assistance and automotive navigation systems [2,3] and can make possible the semiautomatic inventory of important urban scene structures, such as traffic signs [4,5], pole-like objects [6] and roadside trees [7,8].

Accurate 3D data also can be used in community planning [9], map updating [10], and change detection [11,12]. However, the amount of data collected by MLS from a road environment is huge (e.g., more than 100 million points/km). To utilize these data, it needs to be processed efficiently. The automatic classification of 3D MLS point clouds is an important and classical problem in the fields of remote sensing, photogrammetry, computer vision and robotics. It plays the key role in overall point cloud processing work flow because it can contribute to the subsequent use in object recognition, object extraction and scene reconstruction.

However, accurately classifying objects in urban road environments is a difficult task and there are still many problems that remain to be solved. Compared with airborne laser scanning (ALS) point clouds, MLS will get point cloud data with greater densities, which means that some small objects, such as telegraph poles, street lights, curb, etc., can possibly be automatically classified. Simultaneously extracting useful features for both big and small objects from MLS point clouds remains a big challenge for the MLS point cloud classification. Noise, occlusion caused by obstructions and density variation caused by different distances of objects from the sensors are also unavoidable and difficult problems in the MLS point clouds classification. It is necessary to find a superior pipeline and some optimization post-processing strategies to effectively solve the classification problems caused by the incomplete and noisy point cloud data. In addition, even if the MLS point cloud classification has drawn considerable attention, the public data sets with ground truth class labels are still relatively rare. Researchers often need to manually label the original point cloud data, which not only greatly extends the time required for the whole experiment, but also increases the difficulties in comparing the classification precision and recall rate with other methods on the same dataset. Thus, a public dataset that can be used for direct MLS point cloud classification is very necessary.

After analyzing and summarizing various automatic methods for MLS point cloud classification in road environments, this paper proposes an effective overall work flow for the classification of unstructured 3D point clouds. At first, an efficient segmentation approach is used to segment 3D point clouds, which can help to eliminate some noise. In this paper, we employ a point clouds segmentation approach to obtain the original segments. This segmentation approach is a novel hierarchical clustering algorithm named Pairwise Linkage (P-Linkage). Then we propose a two-step post-processing method to obtain a more complete segments. Secondly, we extract a set of features from each segment for training and testing by using a classifier. Thirdly, the contextual constraints among objects is used to refine the classification results based on segments via graph cuts. Experimental results on 3D urban point clouds acquired by a vehicle LiDAR system illustrate that our proposed classification framework is effective.

The main contributions of our work are summarized as follows:

- A unified framework is proposed for classifying the MLS point clouds based on segments.
- A set of features for each segment are well-designed, which can be used to effectively distinguish nine common object classes in urban street scenes.
- A graph cuts energy minimization algorithm using contextual constraints among objects is performed on the initial labeling of the 3D scene to improve the classification precision of the point clouds in urban environments.
- A publicly available point cloud dataset with ground truth is provided for further point cloud classification study.

The remaining part of this paper is organized as follows. After summarizing related works in Section 2, we describe our proposed novel point cloud classification framework in Section 3. Subsequently, in Section 4, we demonstrate the performances in precision, recall rate and efficiency of our proposed classification framework on our built point cloud dataset representing an urban road environment. After that, a discussion about our proposed framework is given in Section 5 followed by a conclusion drawn in Section 6.

2 Related Works

In last decades, a lot of methods have been proposed to solve the point cloud classification problem. Generally, the methods used to classify 3D point cloud acquired from laser scanning data can be divided into two main categories according to the basic elements employed in classification: the point-based classification and the segment-based classification.

The point-based methods classify the 3D point clouds by analyzing the characteristics of single point. For instance, Aparajithan and Shan [13] classified ground points from raw LiDAR data for bald ground Digital Elevation Model (DEM) generation in urban areas by labeling a single point as either a ground one or a non-ground one. Guo *et al.* [14] found that the multiple returns and echoes of the laser pulse can be discriminant features for a single point used to distinguish adjacent objects. In addition to the reflectance and return count information, the point-based point cloud classification methods often calculate some local statistical features in the neighborhood of each individual point [15–18]. However, due to the variation in local 3D structures and point densities, a fixed size of neighborhood can not obtain satisfactory results. Many studies focused on the neighborhood selection methods. Generally, common neighborhood forms are spherical neighborhood [16,19], cylindrical neighborhood [20], voxel neighborhood [21], *K*-nearest neighborhood [22] and combination of multi-scale and various neighborhood shapes [23,24]. Among them, although the methods that combine multi-scale and various neighborhood selection methods can obtain good classification accuracy, this kind of methods don't fundamentally solve the problem caused by uneven density distribution and incompleteness of point clouds. In addition, repetitive calculations of eigenvectors and eigenvalues for each point are required, which greatly increase the computational complexity.

When dealing with large 3D data sets, the computational cost of processing all individual points is very high, making it impractical for real time applications. Besides, those point-based methods maybe fail in some complex point cloud classification conditions due to the limitation of features extracted at the point scale. Therefore, for complex classification consisting of multiple types of objects, many methods segment the original point clouds into voxel or object candidates at first. Then, a set of features that describe, for example, the size and shape of the segment are calculated for each segment, based on which the segments are classified into two or multiple classes. For example, Aijazi *et al.* [25] clustered individual 3D points together to form a voxel level representation. The methods presented in [26–31] tried to segment the original point clouds in the object level. These methods not only can solve the slow computational efficiency problem resulting from the increasing amount of point cloud data, but also extract richer information than the point-based methods. In addition, the segmentation can help removing some noisy points by setting the threshold for the segment size. Thus, the segment-based classification has drawn more attentions in recent years.

In order to obtain a better understanding of the scene and capability of autonomous perception, many methods have proved that effective segmentation is the key to success in the next classification process. Surface discontinuities are widely used in point clouds segmentation, which can be used to segment two adjacent points. For instance, the method presented in [32] used only local surface normals and point connectivity to segment the industrial point clouds and performed well. For urban scenes, other surface features, such as normals, curvatures and the height differences, were widely used to find the smoothly connected areas [33–35]. Segmentation based on individual points may be carried out very efficiently, but there is a severe drawback, namely the noisy appearance of the segmentation results [36]. Many algorithms applied graph cuts [37,38] and Markov random fields [39] to generate the smoother segments than traditional region-growing methods via using neighborhood smoothing constraints. The basic idea of those methods is to first construct a weighted graph where each edge weight cost represents the similarity of the corresponding segments, and then find the minimum solution in this graph. The *k*-Nearest Neighbors (*k*-NN) [40] algorithm is often used to build the graph to improve the efficiency. But the limitation of these methods is that it requires prior

knowledge of the location of the objects to be segmented. Actually, a single point cloud segmentation method will typically not provide a satisfactory segmentation due to the complex geometries and visual appearances in urban scenes. As all points of a segment will obtain the same class label, any under-segmentation will lead to classification errors. Researchers usually adopt the hierarchical segmentation method. The method presented in [36] produced the result of over-segmentation at first, then some post-processing strategies were used to merge the over-segmented parts, and finally it can achieve the results that the same segment contains the objects with a same class as much as possible without the phenomenon of under-segmentation.

After the segmentation, certain classification algorithm is employed to assign each segment with an unique class label. Traditionally, the point cloud classification is completed by manually defining a series of discriminant thresholds to distinguish points for each class. For example, Yu *et al.* [41] segmented the point clouds at first, and then established a hierarchical decision tree to classify each segment into ground, buildings, traffic signs, parterres, trees and others. However, the rules for classification are difficult to be manually set in many cases. To solve this problem, the machine learning method can be applied to learn the classification rules automatically from training data [42]. Firstly, the features of each segment are extracted. For example, Lehtomäki *et al.* [43] applied features describing the global shape and the distribution of the points in an object, such as local descriptor histograms (LDHs) and spin images, in the classification of typical roadside objects. Some methods use the features recorded by scanner systems, such as the reflectance intensity, return count and color information [25,44]. In addition, height-related features, geometrical shape features, eigenvalue-based features, point type, density and orientation are widely used in the state-of-the-art methods [25,28,45–48]. Then a classifier is used to learn the discriminant rules automatically. For example, in outdoor urban scenes, the researchers used Support Vector Machine (SVM) to distinguish basic categories, such as buildings, ground and vegetation [49,50]. In addition, the Random Forests (RF) algorithm was also successfully applied to the LiDAR feature selection to classify urban scenes in [51]. Moreover, the AdaBoost algorithm formed a strong classifier by using simple geometrical features extracted from single laser range scan to classify the points into several semantic classes, like rooms, hallways, corridors, and doorways [52].

Most of the aforementioned classifiers just take into account local features to complete the point classification and ignore the topological relationships between different objects usually existed in urban environments. Thus, it is an effective way to improve the accuracy of classification results by integrating the contextual information into the machine learning framework. The classification approach of a LiDAR point cloud based on Conditional Random Fields (CRF) successfully obtained three basic object classes: vegetation, building and ground [53,54]. Combining CRF with the random forests classifier can obtain more reliable classification results, especially the number of confusions between buildings and larger trees reduced obviously [55]. Moreover, the Associative Markov Network (AMN) was widely used to classify 3D point cloud by utilizing contextual information [56,57].

In this paper, we propose a three-stage classification framework for 3D point cloud classification in the road environment. We make full use of the advancement of segment-based point cloud classification, such as higher computational efficiency and richer information than the point-based methods. In addition, the machine learning methods, such as linear SVM, RF and Extreme Learning Machine (ELM), are used to classify point clouds based on the features extracted from segments. Besides, in order to apply contextual constraints which may not fully performed in the classification procedure, we employ a post-processing procedure by using graph cuts to optimize the initial classification results.

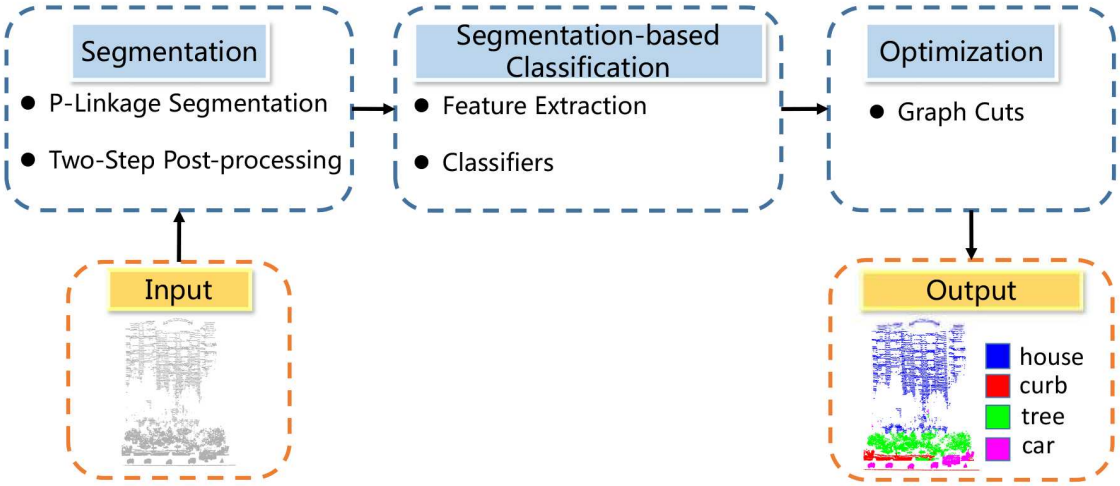


Figure 1. The overview flowchart of our proposed unified framework for classifying 3D urban points clouds acquired in the road environment.

3 Our Approach

In this section, we will give detailed description of our proposed point cloud classification framework. The overall work-flow can be separated into three stages, as illustrated in Figure 1. The first stage is to segment the original unstructured 3D point cloud by using the P-Linkage algorithm, which is a recently proposed region-growing-based hierarchical segmentation algorithm [35]. After that, in order to solve the problems caused by over-segmentation, such as the reduction of the quality of the segment features and the increment of noise, a two-step post-processing approach is proposed: 1) the segments of cars, trees and curbs are grouped by the connected component analysis; 2) nearly co-linear segments of electronic wires and telegraph poles are merged. In the second stage, we extract a set of features from each segment for training and testing by using an effective classifier. For comparison, we employ three classifiers (SVM, RF and ELM) to classify the point clouds, respectively. In the third stage, due to that the classifier cannot give smooth and high accurate results, we use the contextual constraints among objects via the graph cuts energy minimization algorithm to further refine the initial classification.

3.1 3D Point Cloud Segmentation

The key to the success of a segmentation-based classification is, of course, the segmentation. In the case of under-segmentation, points belonging to different object classes will be divided into the same segment. As all points of one segment will obtain the same class label, any under-segmentation will lead to classification errors. The contrary situation, over-segmentation, however, will seriously reduce the quality of the segment features and may lead to some man-made "noise". Furthermore, segment shape descriptors designed specifically for a certain class may become less useful. Therefore, in order to achieve a superior result, people always over-segment a point cloud at first, and then apply some proper post-processing strategies to merge the segments belonging to the same class before classifying segments. In this paper, we firstly over-segment the original unstructured 3D point clouds by using the P-Linkage algorithm. Then we propose a two-step post-processing approach to improve the original segmentation results. Detailed descriptions of the P-Linkage and the post-processing method are presented in the following.

3.1.1 P-Linkage Based Segmentation

The P-Linkage point cloud segmentation algorithm is based on clustering analysis and contains four steps: normal estimation, linkage building, slice creating and slice merging.

1) Normal Estimation: The normal for each point is estimated by fitting a plane to some neighbouring points. The K nearest neighbors (KNN) based method is employed to find the neighbours of each data point and estimate the normal of the neighbouring surface via the Principal Component Analysis (PCA), which is implemented via the ANN library [58] as follows. Firstly, for each data point $\mathbf{p}_i$, its covariance matrix is formed by the first K data points in its KNN set as follows:

$$\mathbf{\Sigma} = \frac{1}{K}\sum_{i=1}^{K}(\mathbf{p}_i - \bar{\mathbf{p}})(\mathbf{p}_i - \bar{\mathbf{p}})^T, \tag{1}$$

where $\mathbf{\Sigma}$ denotes the 3×3 covariance matrix and $\bar{\mathbf{p}}$ represents the mean vector of the first K data points in the KNN set. Then, the standard eigenvalue equation:

$$\boldsymbol{\lambda}\mathbf{V} = \mathbf{\Sigma}\mathbf{V} \tag{2}$$

can be solved using Singular Value Decomposition (SVD), where $\mathbf{V}$ is the matrix of eigenvectors (Principal Components, PCs) and $\boldsymbol{\lambda}$ is the matrix of eigenvalues. The eigenvectors $\mathbf{v}_2$, $\mathbf{v}_1$, and $\mathbf{v}_0$ in $\mathbf{V}$ are defined according to the corresponding eigenvalues sorted in the descending order, i.e., $\lambda_2 \geqslant \lambda_1 \geqslant \lambda_0$. The third PC $\mathbf{v}_0$ is orthogonal to the first two PCs, and approximates the normal $\mathbf{n}(\mathbf{p}_i)$ of the fitted plane. λ_0 estimates how much the points deviate from the tangent plane, which can represent the flatness $\lambda(\mathbf{p}_i)$ of the data point $\mathbf{p}_i$. Finally, the Maximum Consistency with the Minimum Distance (MCMD) algorithm [59] is employed to filter out the outliers neighbouring points for each point cloud, and the inlier neighbouring points is denoted as the Consistent Set $CS(\mathbf{p}_i)$ of the data point $\mathbf{p}_i$. Thus for each data point $\mathbf{p}_i$, we obtain its normal $\mathbf{n}(\mathbf{p}_i)$, flatness $\lambda(\mathbf{p}_i)$ and Consistent Set $CS(\mathbf{p}_i)$.

2) Linkage Building: With the normals, flatnesses and Consistent Sets of all the data points, the pairwise linkage can be recovered in a non-iterative way, which is performed as follows. For each data point $\mathbf{p}_i$ we search in its CS to find out the neighbours whose flatnesses are smaller than that of $\mathbf{p}_i$ and choose the one among them whose normal has the minimum deviation to that of $\mathbf{p}_i$ as $CNP(\mathbf{p}_i)$. If there exists $CNP(\mathbf{p}_i)$, a pairwise linkage between $CNP(\mathbf{p}_i)$ and $\mathbf{p}_i$ is created and recorded into a lookup table $\mathbb{T}$. Otherwise, $\mathbf{p}_i$ is considered as a cluster center, and inserted into the list of cluster centers $\mathcal{C}_{\text{center}}$.

3) Slice Creating: To create the surface slices, the clusters $\mathcal{C}$ are firstly formed by searching along the lookup table $\mathbb{T}$ from each cluster center in $\mathcal{C}_{\text{center}}$ to collect the data points that are directly or indirectly connected with it. Then for each cluster $\mathcal{C}_p$, a slice is created by plane fitting via the MCS method [59] and outlier removing via the MCMD algorithm [59]. Thus for each slice $\mathbf{S}_p$, we obtain its normal $\mathbf{n}(\mathbf{S}_p)$, flatness $\lambda(\mathbf{S}_p)$ and Consistent Set $CS(\mathbf{S}_p)$ in the same way as each data point.

4) Slice Merging: To obtain complete planar and curved surfaces which are quite common in the indoor and industry applications, a normal and efficient slice merging method is proposed. First, we search for the adjacent slices for each one, two slices $\mathbf{S}_p$ and $\mathbf{S}_q$ are considered adjacently if the following condition is satisfied:

$$\begin{aligned} \exists\ \mathbf{p}_i \in CS(\mathbf{S}_p)\ \text{ and }\ \mathbf{p}_j \in CS(\mathbf{S}_q), \\ \text{where }\ \mathbf{p}_i \in CS(\mathbf{p}_j)\ \text{ and }\ \mathbf{p}_j \in CS(\mathbf{p}_i). \end{aligned} \tag{3}$$

Then, for a slice $\mathbf{S}_p$ and one of its adjacent slice $\mathbf{S}_q$, they will be merged if the following condition is satisfied:

$$\arccos\left|\mathbf{n}(\mathbf{S}_p)^\top \cdot \mathbf{n}(\mathbf{S}_q)\right| < \theta, \tag{4}$$

where $\mathbf{n}(\mathbf{S}_p)$ and $\mathbf{n}(\mathbf{S}_q)$ are the normals of $\mathbf{S}_p$ and $\mathbf{S}_q$, respectively, and θ is the threshold of the angle deviation.

3.1.2 Post-Processing

Actually, although the P-Linkage segmentation algorithm can achieve more robust segmentation results than many other methods as described in [35], the general point cloud segmentation method will typically not provide a satisfactory segmentation results for the purpose of classification. Normals for the points near geometric singularities such as edges and corners are usually differently oriented and discontinuous. It may lead to many smooth but non-planar surfaces to be split up into multiple planar patches, such as the segments of trees, cars and curbs. In addition, unexpected interruption resulting from data acquisition and occlusion among different objects may cause discontinuities, such as gaps and holes in the original 3D point cloud data. This phenomenon always appears in buildings, telegraph poles and electric wires which are usually occluded by cars or other objects. To improve the results of initial segmentation, two post-processing steps are proposed to be applied, which are described in detail in the following.

In the first step, we aim to group the broken cars, trees and curbs into the whole ones by using the connected component analysis. The implementation steps are summarized as follows. At first, we find all candidate segments $\mathcal{S}_{\text{candidate}}$ to be merged, which consist of relatively few points and contain enough scatter type points. A segment $\mathbf{S}$ will be picked out as a candidate segment when the number of points is less than the predefined threshold T_b and the ratio between numbers of scatter type points and total points is more than the predefined threshold T_s ($T_b = 500$ and $T_s = 0.5$ were used in this paper), respectively, which are determined by all kinds of factors empirically, such as the sizes of initial segments obtained by the P-Linkage segmentation and the densities of the original point clouds. During the previous segmentation process, for any point $\mathbf{p}$, we can obtain three eigenvalues λ_2, λ_1, and λ_0 ($\lambda_2 \geqslant \lambda_1 \geqslant \lambda_0 > 0$) via PCA which represent the local neighborhood distribution of this point $\mathbf{p}$ in three dimensional space, respectively. The multiple geometric features of the point $\mathbf{p}$ are defined as follows:

$$S_\lambda(\mathbf{p}) = \frac{\lambda_0}{\lambda_2}, \quad L_\lambda(\mathbf{p}) = \frac{\lambda_2 - \lambda_1}{\lambda_2}, \text{ and } P_\lambda(\mathbf{p}) = \frac{\lambda_1 - \lambda_0}{\lambda_2}, \tag{5}$$

where $S_\lambda(\mathbf{p})$, $L_\lambda(\mathbf{p})$, and $P_\lambda(\mathbf{p})$ represent the scatter, linear, and planar geometric features of the point $\mathbf{p}$, respectively. We consider $\mathbf{p}$ as a scatter type point when its scatter geometric feature $S_\lambda(\mathbf{p})$ is higher than the manually set threshold T_λ^s ($T_\lambda^s = 0.1$ was used in this paper). A general region growing strategy is used to merge all the candidate segments $\mathcal{S}_{\text{candidate}}$. Two adjacent segments $\mathbf{S}_i$ and $\mathbf{S}_j$ will be merged only if the minimum Euclidean Distance $d_{\min}(\mathbf{S}_i, \mathbf{S}_j)$ between $\mathbf{S}_i$ and $\mathbf{S}_j$ is less than the predefined threshold T_d ($T_d = 0.3$ was used in this paper). The minimum Euclidean Distance $d_{\min}(\mathbf{S}_i, \mathbf{S}_j)$ is defined as:

$$d_{\min}(\mathbf{S}_i, \mathbf{S}_j) = \min_{\mathbf{p}_k \in \mathbf{S}_i, \mathbf{q}_l \in \mathbf{S}_j} d(\mathbf{p}_k, \mathbf{q}_l), \tag{6}$$

where $d(\mathbf{p}_k, \mathbf{q}_l) = \|\mathbf{p}_k - \mathbf{q}_l\|$ is the Euclidean Distance between $\mathbf{p}_k$ and $\mathbf{q}_l$.

In the second step, we try to merge the co-linear segments, such as the segments of telegraph poles and electric wires. Similar to the first step, we firstly find all the candidate segments $\mathcal{S}_{\text{candidate}}$ to be merged, which have enough linear type points. A segment will be picked out as a candidate segment when the ratio between numbers of linear type points and total points is more than the predefined threshold T_l ($T_l = 0.5$ was used in this paper). We consider $\mathbf{p}$ as a linear type point when its linear geometric feature $L_\lambda(\mathbf{p})$ is higher than the manually set threshold T_λ^l ($T_\lambda^l = 0.75$ was used in this paper). The same region growing method as in the first step is used to merge the candidate segment $\mathcal{S}_{\text{candidate}}$ but with different merging condition for two adjacent segments. Two adjacent segments $\mathbf{S}_i$ and $\mathbf{S}_j$ will be merged if they satisfy the following two conditions: 1) the intersection

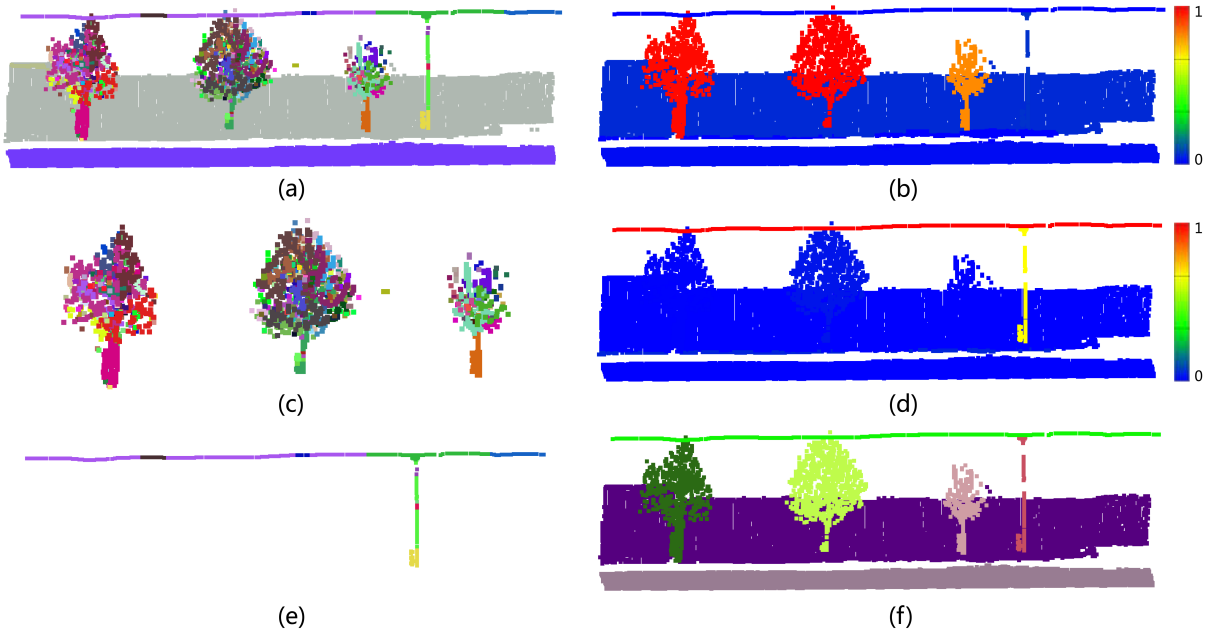


Figure 2. An illustration of our proposed segmentation post-processing strategies: (a) the original P-Linkage segmentation result; (b) the ratios of scatter points for each segment, which range from 0 to 1; (c) the candidate segments in the first-step post-processing; (d) the ratios of linear points for each segment, which range from 0 to 1; (e) the candidate segments in the second-step post-processing; (f) the final segmentation result after two-step post-processing.

angle $\theta(\mathbf{S}_i, \mathbf{S}_j)$ between the direction vectors (the first PC $\mathbf{v}_2$) of the two segments is less than the predefined threshold T_θ ($T_\theta = 30^\circ$ was used in this paper); 2) the Orthogonal Distance (OD) $d(\mathbf{S}_i, \mathbf{S}_j)$ between the direction vectors of two segments is less than the predefined threshold T_{od} ($T_{\text{od}} = 0.3$ was used in this paper). These two specific calculation formulas are defined as follows:

$$\theta(\mathbf{S}_i, \mathbf{S}_j) = \arccos \frac{\mathbf{v}_2(\mathbf{S}_i) \cdot \mathbf{v}_2(\mathbf{S}_j)}{|\mathbf{v}_2(\mathbf{S}_i)||\mathbf{v}_2(\mathbf{S}_j)|} \quad \text{and} \quad d(\mathbf{S}_i, \mathbf{S}_j) = \frac{(\mathbf{v}_2(\mathbf{S}_i) \times \mathbf{v}_2(\mathbf{S}_j)) \cdot \overrightarrow{\mathbf{c}(\mathbf{S}_i)\mathbf{c}(\mathbf{S}_j)}}{|\mathbf{v}_2(\mathbf{S}_i) \times \mathbf{v}_2(\mathbf{S}_j)|}, \tag{7}$$

where the $\theta(\mathbf{S}_i, \mathbf{S}_j)$ and $d(\mathbf{S}_i, \mathbf{S}_j)$ represent the intersection angle and the OD between the direction vectors $\mathbf{v}_2(\mathbf{S}_i)$ and $\mathbf{v}_2(\mathbf{S}_j)$ of the two segments $\mathbf{S}_i$ and $\mathbf{S}_j$, respectively, $\mathbf{c}(\mathbf{S})$ denotes the weighted center point of some segment $\mathbf{S}$, the operators '$\times$' and '$\cdot$' denote the cross and dot products between two vectors, respectively, and $\overrightarrow{\mathbf{c}(\mathbf{S}_i)\mathbf{c}(\mathbf{S}_j)}$ denotes the direction vector from $\mathbf{c}(\mathbf{S}_i)$ to $\mathbf{c}(\mathbf{S}_j)$.

To present our post-processing method clearly, we picked out an example scene as shown in Figure 2, containing parts of trees, street lights, electric wires, fences and the ground. From Figure 2(b), we find all the candidate segments for the first-step post-processing which consist of few points and contain enough scatter type points, as shown in Figure 2(c). From Figure 2(d), we find the candidate segments for the second-step post-processing, as shown in Figure 2(e). Finally, the segmentation result after two-step post-processing will be improved, as shown in Figure 2(f).

Table 1. A list of features extracted from a point cloud segment.

Categories	Features
Orientation	The angle between the normals of the segment and the Z-axis
Heights	The relative height of the segment
	The height standard deviation
Geometrical shapes	The U-V plane projection area
	The U-Z plane projection area
	The V-Z plane projection area
	The ratio between the U-V and U-Z plane projection areas
	The ratio between the U-V and V-Z plane projection areas
	The ratio between the U-Z and V-Z plane projection areas
Point types	The percentage of scatter type points
	The percentage of horizontal type points
	The percentage of slope type points
	The percentage of vertical type points
	The percentage of linear type points
	The percentage of planar type points
Densities	The U-V plane projection density
	The U-Z plane projection density
	The V-Z plane projection density

3.2 Segmentation-Based Classification

3.2.1 Feature Extraction

The classification accuracy highly depends on the qualities of the predefined features. In this work, in order to classify these segments into special object classes accurately, we design five kinds of geometrical and local features based on the characteristics of different types of objects. These features are summarized in Table 1 and described in detail in the following.

1) Orientation: The orientation of the surface is found essentially for the classification between the ground and the building facades. For the ground objects, the surface normals are predominantly along the Z-axis (i.e., the height axis), whereas for building facades, the surface normals are predominantly parallel to the X-Yaxis (i.e., the ground plane). So, at first, the normal vector of the entire segment can be obtained by PCA. Then we calculate the angle between the normal and the Z-axis to obtain the orientation information, which ranges from 0° to 90°. The segments on the ground will get an orientation value close to 0°, while the segments on buildings will get a value at around 90°.

2) Heights: The height-related features are very useful to distinguish the objects with different heights, like buildings, fences, cars and grounds. In our case, we separately calculate two kinds of height-related features. The first one is the relative height of one segment. We use an improved filtering model to find all ground points. This model is based on the traditional sliding window model and is combined with optimized rules on determining the standard elevation value, the tolerance of elevation difference and the dynamic thresholds [60]. Then, for each point of one segment, we obtain the relative height by subtracting the average Z value of its three nearest ground points from the Z value of this point. Finally we acquire the relative height of a segment easily by calculating the average relative height of all the points from the segment. We set the relative height for all ground points as 0. This height-related feature may can not distinguish some objects from classes that have a similar average relative height, for example, trees and electric wires have almost the same relative height in most road environments. Therefore, we employ another height-related feature, the height

standard deviation, to effectively solve this problem and distinguish those different classes with the similar relative height. The definition of this feature is given as follows:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(Z_i - \mu)^2}, \tag{8}$$

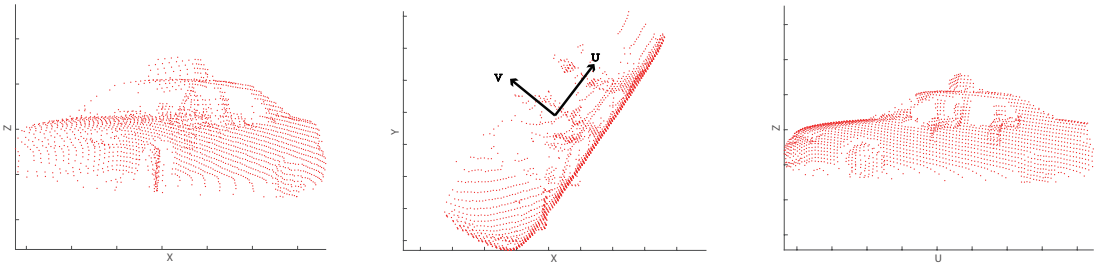
where σ represents the height standard deviation, N represents the point number of the segment, μ represents the average relative height of the segment and Z_i represents the relative height of the i-th point in this segment. The height standard deviation can effectively distinguish the situation in which segments from different classes have a similar average height but have different internal elevation differences. For instance, as mentioned before, the trees and electric wires have almost the same relative height. However, the height differences of the points in the tree segments are relatively larger than those of electric wires. So the height standard deviation feature can be effectively used to separate these two classes.

3) Geometrical shapes: Objects in different classes have different geometrical shapes in general. For instance, the ground can be represented as a low flat plane. The buildings can be represented as large and vertical planes. The minimum bounding box of trees and cars are almost broad and short. Figures 3(a) and (b) show an example of the car class in the X-Z plane (i.e., the front view) and the X-Y plane (i.e., the top view), respectively. From Figure 3, we can find that the car class is diagonal. It is necessary to compare a training example with a testing example from the same point of view. In order to avoid such a rotate-variant property in the X-Y plane, the example is transformed into the U-V axes (i.e., the arrows in the middle of Figure 3(b)), which are the principle axes obtained by PCA. Figure 3(c) shows the example after the transformation in the U-Z plane. By doing so, training and testing examples can be compared from the same point of view without the rotate-variant property.

In total, we calculate two kinds of features related to the object geometrical shape. The first one is the projection area. In order to display the geometrics of the objects in all directions, we calculate 3 kinds of projection areas. We separately project the segment onto the U-V plane, the U-Z plane and the V-Z plane. However, the segment projection area calculation is a quite complex process, especially for those with complex structures. Thus, we can calculate the projection area approximately by the following method, as one example clearly showed in Figure 4. At first, in order to estimate projection area, we establish a regular grid on the projection plane. According to the actual calculative precision, we manually adjust the width of the grid cell. Then if there is at least one point project onto a cell, then this cell can be set as occupied. Finally, we count how many cells have been occupied. The projection area will be obtained by multiplying the number of occupied cells by the area of single cell.

In order to measure the relationship among three projection areas, the second feature related to the geometrical shape is the ratio among the 3 kinds of projection areas. We can get three ratio-related features for each segment: the ratio between the U-V and U-Z plane projection areas, the ratio between the U-V and V-Z plane projection areas, and the ratio between the U-Z and V-Z plane projection areas. Here we require dividing the large projection area by the small projection area, namely, the ratios must be larger than one.

4) Point types: For the purpose of classifying all unknown-class segments according to component difference of each segment, we should perform some data preprocessing like the point type extraction. According to the orientation value as described before, we can divide all the points into 4 categories: scatter type, horizontal one, slope one and vertical one. The calculation of the orientation value and the type extraction in detail are showed in Algorithm 1. Then we respectively calculate four point-type-related features by the means of dividing the point number of the corresponding point type by the number of total points in the segment for 4 different types. The values of these features all range from 0 to 1. The segments from trees have the almost "1" value for scatter type points. The floor segments have almost "1" value for horizontal type points and almost "0" value of vertical type points, while the house and fence segments have the completely opposite



(a) (b) (c)

Figure 3. An example for the rotation-invariant geometrical shape of a car: (a) the X-Z plane (i.e., the front view); (b) the X-Y plane (i.e., the top view) where the arrows represent the principle axes obtained by PCA (i.e., the U-V plane); (c) the U-Z plane.

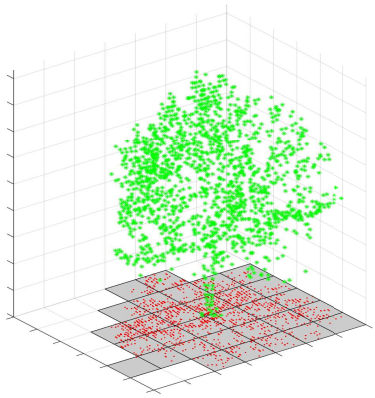


Figure 4. A tree example for the projection area calculation. The green points represent the original point clouds, and the red points are the projected points on the projection plane. The grey cells stand for the occupied areas which are counted to calculate the projection areas.

situations. As for other classes, such as cars, around the doors and the side surfaces, vertical type points appear. Near the window frames, there exist slope type points due to inclined surfaces. And it also has some scatter type points because of some uneven surfaces. For the curbs, most points are slope and vertical types. Even through these four point types can distinguish most categories, there are still some categories difficult to be separated from, especially for telegraph poles and street lights. Therefore we define other 2 features to deal with such situation: linear and planar type features. At first, we calculate all the linear and planar values of all the points from each segment by Eq. (5). Then we calculate the average value of each segment.

5) Densities: The nearer the object is far away from the laser point emission center, the larger the point density is. Thus, we re-sample the original point cloud data to make the same number of points per cubic meter. Then, we calculate the projection point densities of three projection directions: the U-V plane, the U-Z plane and the V-Z plane. They can be estimated by using total number of points after re-sampling divided by the projection area of the segment which can be processed by the same ways as we mentioned before. This kind of features can effectively distinguish the classes with a great projected density in one projection direction, such as the telegraph poles, which have a extremely small projection area on the U-V plane resulting in a large density value in this direction.

Algorithm 1 Determining the point type.

Input: The normal $\mathbf{n} = (n_x, n_y, n_z)^\top$, the minimum eigenvalue λ_0, and the maximum eigenvalue λ_2 of a point $\mathbf{p}$.
Output: The point type of $\mathbf{p}$.

if $\lambda_0/\lambda_2 > 0.1$ **then**
  $\mathbf{p}$ is a scatter type point.
else
  Calculating the angle θ between $\mathbf{n}$ and the Z-axis as follows:

$$\theta = \tan^{-1} \frac{|n_z|}{|n_x|^2 + |n_y|^2}. \tag{9}$$

  if $0^\circ \leq \theta < 30^\circ$ **then**
    $\mathbf{p}$ is a horizontal type point.
  else if $30^\circ \leq \theta \leq 60^\circ$ **then**
    $\mathbf{p}$ is a slope type point.
  else if $60^\circ < \theta \leq 90^\circ$ **then**
    $\mathbf{p}$ is a vertical type point.
  end if
end if

3.2.2 Classifiers

After feature extraction, each segment will have a high-dimensional feature vector, and each segment for training will have only one class label. Then, we formulate the classifier as a function of predicting the class label of the segment. As for the comparisons, we apply three popularly used classifiers, SVM, RF and ELM, to validate the extracted features, respectively.

1) SVM classifier: The support vector machine (SVM) is a very popular classifier and has been widely used in many fields of computer vision, which is possible to split apart different types of samples in the high-dimensional space by obtaining the most optimal hyperplanes. In our work, the software package libSVM provided by [61] was applied to automatically complete all the operations including data normalization and parameter selection.

2) RF classifier: The random forests (RF) [62] is a combination of tree predictors. It has excellent performance in classification tasks compared with many other machine learning classifiers, sometimes even better than SVM. The RF is also widely used in the remote sensing data classification. But in the field of point cloud classification, most of researches applied the RF to classify the airborne laser point clouds, and rarely applied it to classify the mobile laser scanner data. In our work, we apply the RF to classify the mobile point clouds captured form urban road scenes.

3) ELM classifier: The extreme learning machine (ELM) [63] is a single-hidden layer feedforward neural networks (SLFNs) which randomly chooses hidden nodes and analytically determines the output weights of SLFNs. It tends to ensure the high accuracy of learning results at extremely fast learning speed. The ELM has not yet been used to classify the mobile laser scanner data.

3.3 Optimization via Graph Cuts

To refine initial classification results and achieve more smooth and accurate ones, we formulate this problem as an energy optimization problem and solve it via graph cuts. We can refine the class labels of the small and possibly misclassified objects as those of their nearest and reliably classified objects by such an optimization.

At first, the region growing algorithm is used to cluster the initial classification result. Similar to the region growing strategies described in Section 3.1.2, two neighboring segments $\mathbf{S}_i$ and $\mathbf{S}_j$ will be merged if the following two conditions are satisfied: 1) the minimum Euclidean Distance

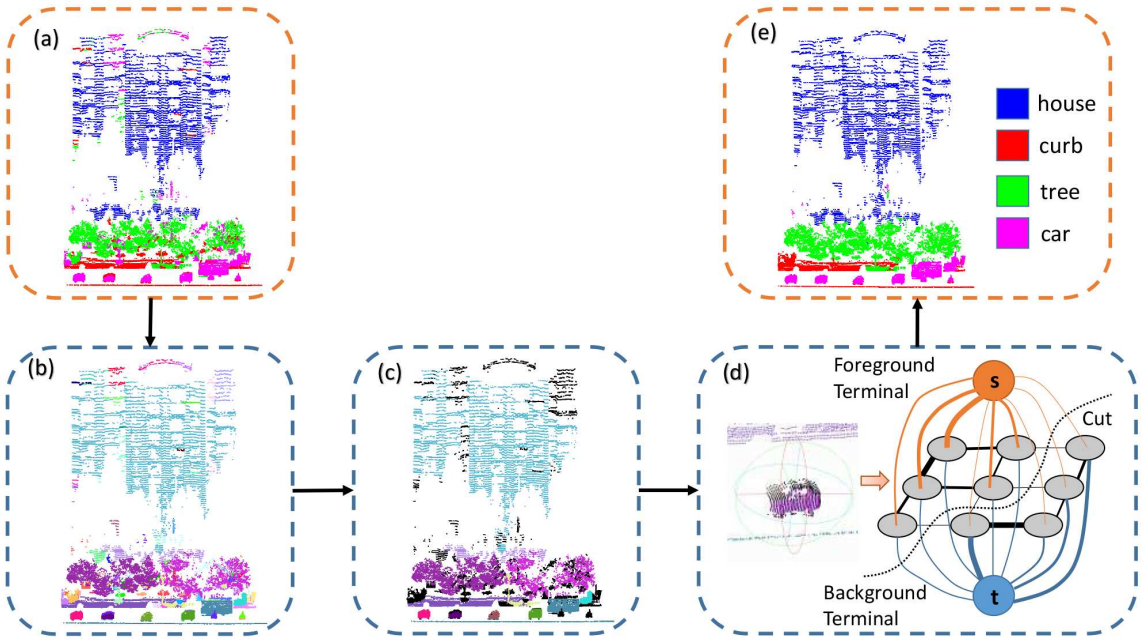


Figure 5. An illustration of classification refinement via graph cuts: (a) the initial classification results; (b) the region growing result based on the same initial classification labels where different colors represent different objects; (c) the separation of reliable and unreliable objects according to their point sizes where unreliable objects are represented by black points; (d) the classification refinement of unreliable points for each reliable object by applying the graph cuts optimization; (e) the finally refined classification results after optimization.

$d_{\min}(\mathbf{S}_i, \mathbf{S}_j)$ between them is less than the threshold T_d ($T_d = 0.3$ was used in this paper); 2) the class labels of $\mathbf{S}_i$ and $\mathbf{S}_j$ are the same. After that, the merged individual objects will be recognized as the reliable and unreliable ones according to the numbers of the points in these objects. For different object classes, different number thresholds are used to separate objects in each object class into the reliable and unreliable ones. For example, the thresholds of buildings and the ground are relatively large, while the threshold of cars is small. For each reliable object associated with its neighbouring points, the graph-cuts-based foreground/background segmentation is successively used to refine the classification results, and the flowchart is illustrated in Figure 5. The data range for the currently selected reliable object $\mathbf{O}_r$ to be optimized is a 3D spherical region whose center is the gravity point of $\mathbf{O}_r$ (i.e., $\mathbf{c}(\mathbf{O}_r) = 1/|\mathbf{O}_r| \sum_{\mathbf{p}_k \in \mathbf{O}_r} \mathbf{p}_k$, where $|\mathbf{O}_r|$ denotes the number of the points in $\mathbf{O}_r$) and whose radius is the maximum Euclidean Distance between the center $\mathbf{c}(\mathbf{O}_r)$ and the points in the current reliable object, i.e., $\max_{\mathbf{p}_k \in \mathbf{O}_r} \|\mathbf{c}(\mathbf{O}_r) - \mathbf{p}_k\|$. The radius of the 3D spherical region is gradually expanding until it contains not only some points in other reliable objects, but also some points in some unreliable objects. For all points in $\mathbf{O}_r$, we consider them as the foreground seed set $\mathcal{F}$. The points in other reliable objects in the 3D spherical region are considered as the background seed set $\mathcal{B}$. The class labels of the foreground and background seed points are fixed. All the points from the unreliable objects are regarded as the points whose class labels need to be optimized via graph cuts, which is denoted as the set $\mathcal{U}$.

To implement the optimization, we construct a graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ where $\mathcal{V}$ consists of all points $\mathcal{P} = \mathcal{U} \bigcup \mathcal{F} \bigcup \mathcal{B}$ in the spherical region and two terminal points $\mathbf{s}$ and $\mathbf{t}$ which are the gravity points of the foreground seed sets $\mathcal{F}$ and $\mathcal{B}$, respectively, i.e., $\mathcal{V} = \mathcal{P} \bigcup \{\mathbf{s}, \mathbf{t}\}$. The edge set $\mathcal{E}$ consists of two types of undirected edges: n-links (neighborhood links) and t-links (terminal links). Each point $\mathbf{p}$ in $\mathcal{P}$ has two t-links $\{\mathbf{p}, \mathbf{s}\}$ and $\{\mathbf{p}, \mathbf{t}\}$ connecting it to each terminal. Any pair of neighboring points $\{\mathbf{p}, \mathbf{q}\}$ in $\mathcal{P}$ is connected by a n-link and all these n-links constructed from the points in $\mathcal{P}$ is denoted

Table 2. The settings for weights of edges in graph cuts.

Types of Edges	Edges	Weights	Conditions
n-**links**	$\{\mathbf{p},\mathbf{q}\}$	$w(\mathbf{p},\mathbf{q})$	$\{\mathbf{p},\mathbf{q}\}\in\mathcal{N}$
t-**links**	$\{\mathbf{p},\mathbf{s}\}$	$w(\mathbf{p},\mathbf{s})$	$\mathbf{p}\in\mathcal{U}$
		$\max_{\mathbf{q}\in\mathcal{U}} w(\mathbf{q},\mathbf{s})$	$\mathbf{p}\in\mathcal{F}$
		0	$\mathbf{p}\in\mathcal{B}$
	$\{\mathbf{p},\mathbf{t}\}$	$w(\mathbf{p},\mathbf{t})$	$\mathbf{p}\in\mathcal{U}$
		0	$\mathbf{p}\in\mathcal{F}$
		$\max_{\mathbf{q}\in\mathcal{U}} w(\mathbf{q},\mathbf{t})$	$\mathbf{p}\in\mathcal{B}$

as the set $\mathcal{N}=\{\{\mathbf{p},\mathbf{q}\}|\mathbf{p},\mathbf{q}\in\mathcal{P},\mathbf{p}\neq\mathbf{q}\}$. Thus, the edge set $\mathcal{E}=\mathcal{N}\bigcup\{\{\mathbf{p},\mathbf{s}\},\{\mathbf{p},\mathbf{t}\}|\mathbf{p}\in\mathcal{P}\}$. The weights of edges in $\mathcal{E}$ are calculated according to Table 2, where the weight value of some edge $\{\mathbf{p},\mathbf{q}\}$ connecting any two nodes $\mathbf{p}$ and $\mathbf{q}$ is calculated as follows:

$$w(\mathbf{p},\mathbf{q})=\exp\left(-\frac{d(\mathbf{p},\mathbf{q})^2}{\sigma^2}\right), \tag{10}$$

where $d(\mathbf{p},\mathbf{q})=\|\mathbf{p}-\mathbf{q}\|$ is the Euclidean Distance between $\mathbf{p}$ and $\mathbf{q}$, and σ is the average weight of all the edges in $\mathcal{E}$, i.e., $\sigma=\frac{1}{|\mathcal{E}|}\sum_{\{\mathbf{p},\mathbf{q}\}\in\mathcal{E}} d(\mathbf{p},\mathbf{q})$. Finally, we find the best solution via graph cuts by segmenting the points into the classes of foreground or background. After optimization, the class labels of unreliable points that are segmented to the foreground seed set $\mathcal{F}$ are adjusted to the label of the corresponding reliable object in $\mathcal{F}$, but the labels of all other points segmented to the background seed set won't be changed.

4 Results

Our proposed algorithm was implemented and tested in a computer Intel(R) Core(TM) i7-4770 CPU at 3.4GHz and the 8 GB RAM memory. In order to improve the computational efficiency, the parallel computing technique was employed. To validate the performance of our proposed algorithm, we applied both qualitative and quantitative evaluations on our own built dataset.

4.1 Dataset

The point cloud dataset used in this paper was captured form Huangshi city of Hubei Province in China, and this data was acquired by using the SICK LMS511 laser scanners mounted on a vehicle. The total length of this original data is 33.5km and the size of this data is 11.7GB. We only selected partial data to conduct our experiments. We constructed the ground truth dataset by manually labeling each 3D point with corresponding object class. All of these operations were implemented by using an open source software *CloudCompare* [1], which is a 3D point cloud and mesh processing software. By observing the original data, we choose the following nine class labels: ground, buildings, cars, trees, curbs, fences, street lights, telegraph poles and electric wires. The dataset finally consists of 7 separately continuous point clouds. The informations of the dataset are presented in Table 3, which is publicly available for downloading at http://cvrs.whu.edu.cn.

[1] Available at http://www.cloudcompare.org/.

Table 3. The informations of manually labeled class ground truth point cloud dataset.

Sets	#Points	Ground	Buildings	Cars	Trees	Curbs	Fences	Street Lights	Telegraph Poles	Electric Wires
S_1	1,050,774	447,822	257,125	18,527	260,738	22,369	34,199	3,596	3,979	2,419
S_2	1,074,792	561,797	177,267	13,206	125,464	1,633	186,343	1,913	4,778	2,391
S_3	975,256	497,100	137,812	8,526	207,427	31,429	80,787	2,907	3,301	5,967
S_4	724,598	377,269	129,863	9,879	129,543	29,026	37,582	1,181	5,712	4,543
S_5	713,367	309,787	102,062	10,898	247,539	36,597	1,070	1,921	2,962	531
S_6	1,239,388	595,236	355,448	18,288	255,966	1,122	1,274	1,598	7,083	3,373
S_7	1,452,821	772,333	353,405	28,944	241,177	0	43,630	2,023	7,886	3,423

Note: The overall length of the data set is about 5643 meters.

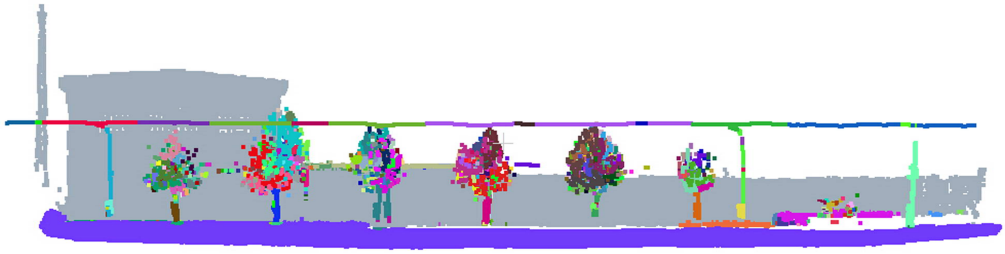
Table 4. Segmentation statistical results of 3 sets of point clouds.

Sets	#Points	#Segments via P-Linkage	#Segments After Post-processing
S_1	1,050,774	40,718	667
S_2	1,074,792	21,083	479
S_3	975,256	30,714	558

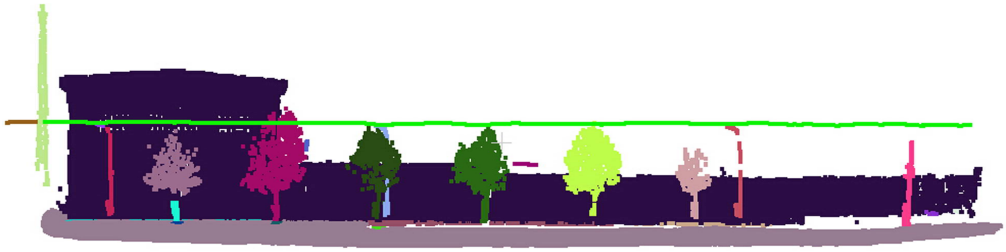
4.2 Segmentation

The segmentation of dense laser scanner points acquired by a vehicle-mounted platform is a challenging task due to the existence of varied kinds of road furnitures which contain grounds, building facades, cars, trees, curbs, fences, street lights, telegraph poles and electric wires. To test the robustness of the proposed point cloud segmentation method, we applied it on 3 sets of laser scanner point clouds (S_1, S_2 and S_3 as shown in Table 3) from our built dataset, which consist of $1,050,774$, $1,074,792$ and $975,256$ points, respectively. These point clouds are unstructured and only supply 3D coordinate information without corresponding RGB color values and reflectance intensities. In all these tests, we only adjusted the value of the parameter θ for slice mering (see Eq. (4)) to get the best P-Linkage segmentation results and set the minimum point number of each segment in order to make a simple noise filtering effect. In our experiments, we uniformly set the value of θ as 15° and the minimum point number for each valid segment empirically is set as 10.

The detailed informations and segmentation results are summarized in Table 4. After post-processing, the number of segments is obviously less than one of segments obtained by P-Linkage. In order to show the segmentation results clearly, Figure 6 and Figure 7 presented the segmentation results of two regions before and after post-processing, respectively. These two regions were selected from the point cloud set S_2, which are named as region I and region II, respectively. From both Figure 6 and Figure 7, we observed that the road surfaces were clustered into a complete one separated entirely from other objects. Besides, the building facades were segmented quite completely and well despite that their densities vary in a wide range. However, except for planar objects, such as building facades, walls and floors, the other linear objects and curved-surface objects are always over-segmented. For example, from Figure 6(a), we can find that a complete electric wire and some completely individual trees were segmented into multiple small segments after applying the original P-Linkage segmentation algorithm. In addition, from Figure 7(a), we observed that sometimes the over-segmentation problem also appears in the regions of cars, telegraph poles and street lights. However, after the post-processing operation, all those problems mentioned above have been greatly solved. As shown in Figure 6(b) and Figure 7(b), most of the original segments of the same object were merged together and our proposed post-processing strategies greatly solve



(a) the original P-Linkage segmentation results for the region I from the point cloud set S_2.



(b) the segmentation result after post-processing for the region I from the point cloud set S_2.

Figure 6. The segmentation results before and after post-processing for the region I from the point cloud set S_2.

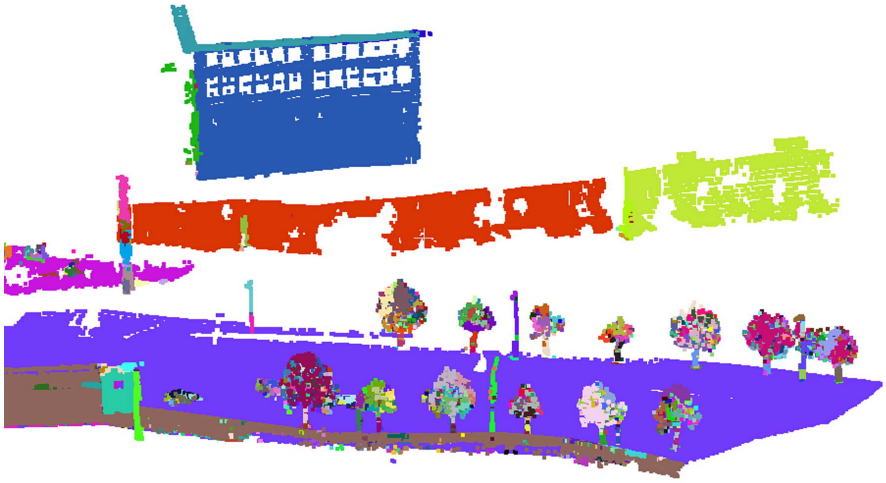
the over-segmentation problem. More accurate segments will effectively improve the accuracy of the next segmentation-based classification.

4.3 Initial Classification

The dataset with seven sets of point clouds was divided into two parts: training dataset and testing dataset. The first three sets S_1, S_2 and S_3 were used as the testing dataset. The remaining four sets were used to train the classifiers. The trained classifiers were then used to classify the testing dataset.

We conducted two comparative experiments to verify the effectiveness of our proposed approach. To validate the effectiveness of our proposed post-processing strategies for classification, in the first experiment, we compared the classification results based on the segments before and after post-processing on the three testing point cloud sets using the SVM classifier, as shown in Table 5. In default, the SVM classifier was used in all the following experiments unless clearly stated. It can be seen from Table 5 that better classification performances (with higher precisions and recalls) can be achieved based on the segments after applying our proposed post-processing strategies than ones based on the segments obtained by the original P-Linkage segmentation algorithm. On average, the precisions and recall rates increase by about 10.7% and 14.4% on the three testing sets using the segments obtained after post-processing, respectively.

In addition, in order to prove that our extracted features are also applicable to other classifiers except for SVM, we also conducted a comparative experiment among three different kinds of classifiers: SVM, RF and ELM. In this experiment, we used the same segmention results and the same features associated with different kinds of classifiers. Both RF and ELM classifiers were all implemented in Matlab. Through a series of comparative experiments, we obtained the best set of parameters for RF and ELM classifiers, respectively. As for RF, the number of trees is 200 and the number of splits for each tree node is 15. As for ELM, the number of hidden neurons is 100 and the activation function type is the sigmoid function. The classification performances for three different classifiers on the three testing sets are showed in Table 6. On average, the SVM classifier reaches a



(a) the original P-Linkage segmentation results for the region II from the point cloud set S_2.

(b) the segmentation result after post-processing for the region II from the point cloud set S_2.

Figure 7. The segmentation results before and after post-processing for the region II from the point cloud set S_2.

Table 5. The comparison results between the classification of original P-Linkage segmentation and the classification based on the segmentation after post-processing for 3 testing point clouds.

Sets	Original P-Linkage Segmentation		P-Linkage+Post-Processing	
	Precision (%)	Recall (%)	Precision (%)	Recall (%)
S_1	72.8	62.7	87.4	79.1
S_2	79.3	63.0	92.9	77.5
S_3	76.8	65.4	80.8	77.6

precision of 87.0% and a recall rate of 78.1%. The RF classifier reaches a precision of 87.9% and a recall rate of 77.6%. The ELM classifier reaches a precision of 76.9% and a recall rate of 71.1%. On average, both SVM and RF classifiers reach the similar precisions and recall rates, which are higher than ones of the ELM classifier.

The under-segmentation appears when two objects are too close to each other, which will greatly decrease the classification performance. For example, many street light points were mistakenly

Table 6. The classification performances for the three testing point cloud sets using three kinds of classifiers (SVM, RF and ELM).

Sets	Precision (%)			Recall (%)		
	SVM	RF	ELM	SVM	RF	ELM
S_1	87.4	86.9	71.4	79.1	74.8	72.2
S_2	92.9	84.6	81.3	77.5	76.6	66.6
S_3	80.8	92.1	78.0	77.6	81.5	74.6

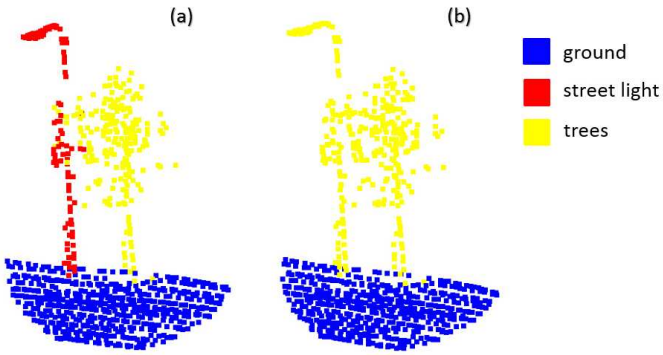


Figure 8. A misclassification example in which the street light points were mistakenly classified as trees due to under-segmentation: (a) the ground truth; (b) the original classification result.

labeled as trees because the street lights are too close to the trees and the parts of the street lights are hidden in the trees, as shown in Figure 8. There are also many misclassified points among trees, buildings and fences. Due to occlusion and other reasons, sometimes the building and fence surfaces are not too smooth as planes which can be easily merged with the trees during segmentation and post-processing. These under-segmentation errors also appear frequently when the trees clings to the front of the fences, as shown in Figure 9. In general, the point clouds for the buildings obtained by a vehicle LiDAR system are often incomplete and finely broken, which will result in over-segmentation. In this case, the small segments from buildings were often misclassified into other object classes, such as cars, trees, etc, as shown in Figure 10.

4.4 Optimization Evaluation

By the observation of the original classification results, we found that many classification errors result from over-segmentation as shown in Figure 10. Since objects themselves were finely broken and incomplete, there exist always some gaps inside the objects. Our proposed post-processing strategies will not solve such this over-segmentation efficiently and completely. To reduce the misclassification caused by over-segmentation, we adopted the graph cuts method to optimize the initial classification results. Firstly, we grouped the points with the same initial classification labels and close distances. Secondly, according to the point numbers of objects, all the targets were divided into reliable and unreliable ones. Then we built a graph model for each reliable object. All of the small and unreliable objects around the reliable object were merged into the reliable object via graph cuts. Finally, we will get a smoother and more refined classification result. Figure 11 shows some close-ups of the final classification results after applying the graph cuts optimization algorithm. The generated segmentation result shows clearly distinguished labeling results for different objects. To quantitatively evaluate our proposed optimization approach, we compared the classification performances before and after optimization. The precisions and recall rates of three testing sets for the initial classification and the classification after optimization are reported in Table 7. The

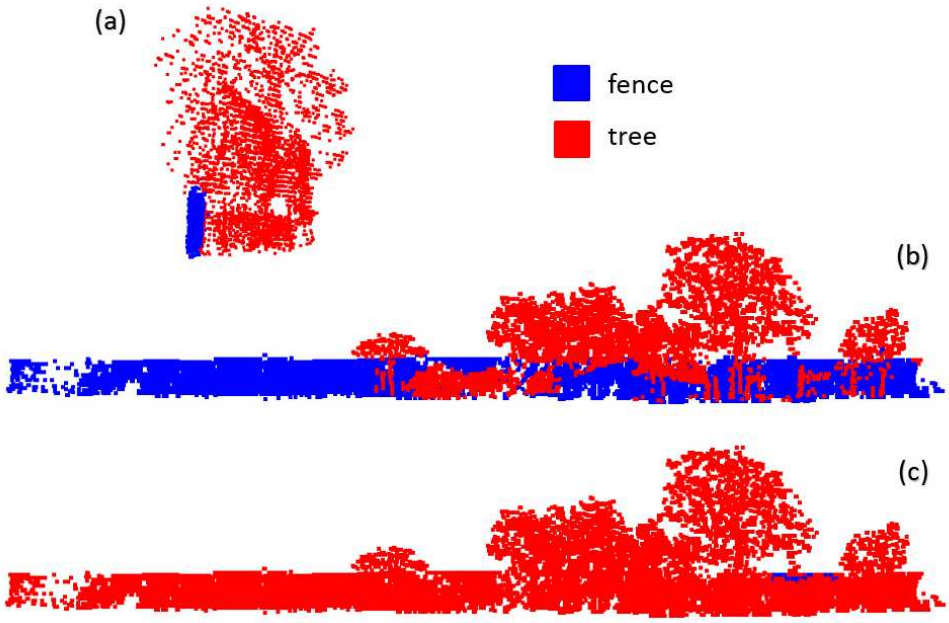


Figure 9. A misclassification example in which the fence points were mistakenly classified as trees due to under-segmentation: (a) the side view of the ground truth; (b) the front view of the ground truth; (c) the classification result.

Figure 10. A misclassification example in which some small segments from buildings were mistakenly classified as other object classes such as cars and trees due to over-segmentation.

confusion matrix of all the three testing sets for the initial classification and the classification after optimization are reported in Table 8 and Table 9, respectively. As the misclassification results from over-segmentation and the high similarity between certain object classes are inevitable , the classification results can not reach a very high precision. However, by comparing the results before and after optimization, we can observe that both the precisions and recall rates of different object classes are improved significantly, especially for the recall rates. In particular, the recall rate of the point cloud set S_2 increases by 4.8% after optimization which gets the highest increase in the three testing sets. The overall precisions for final classification reach 81.5%-93.3% and the overall recall rates reach 80.2%-82.3%.

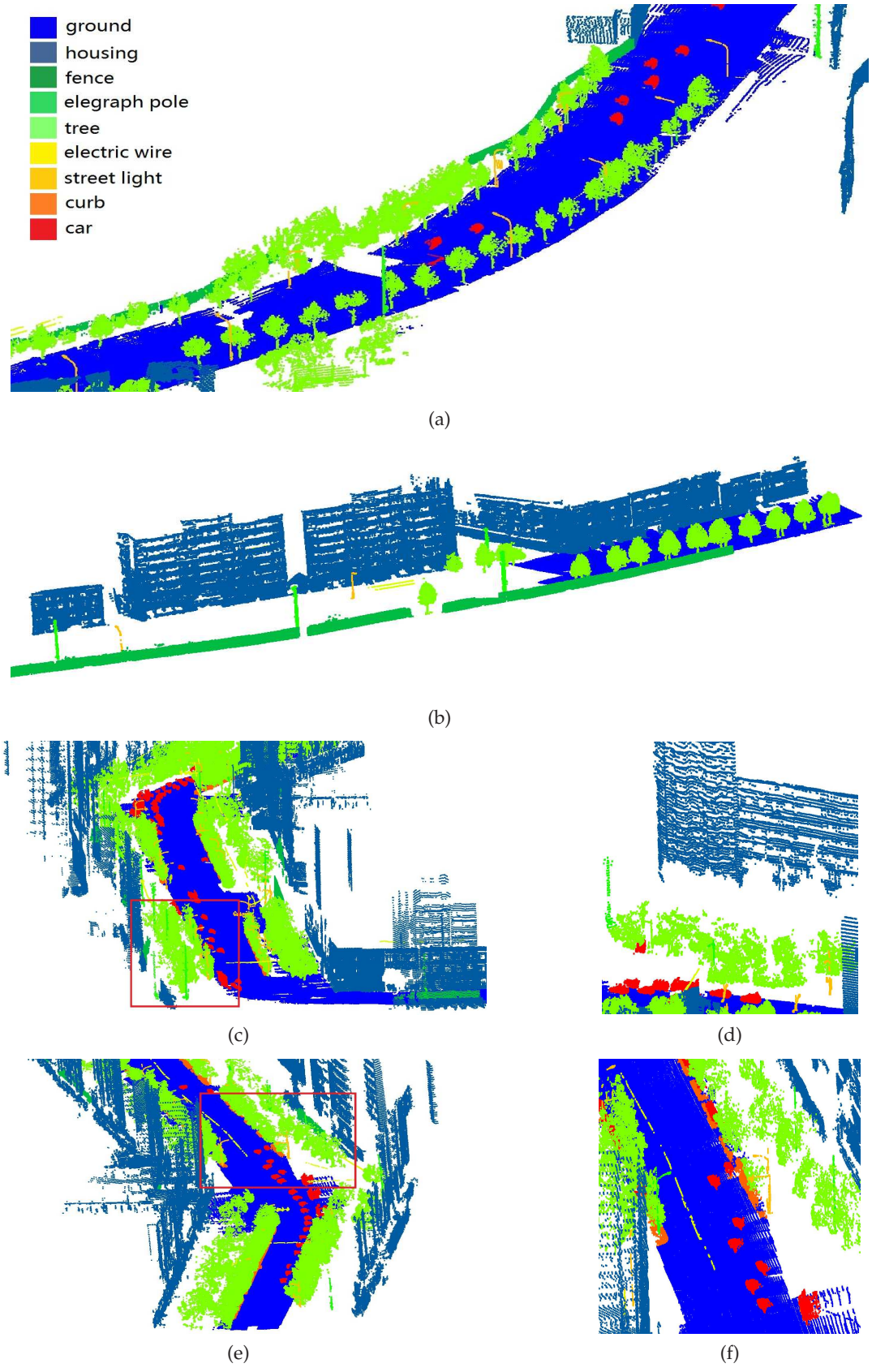


Figure 11. Close-ups of the classification result after optimization: (a) the street view with grounds, buildings, fences, trees, street lights, cars and telegraph poles; (b) the street view with grounds, fences, buildings, trees, telegraph poles, street lights and electric wires; (c) and (e) the street view with all the 9 object classes; (d) and (f) the local details of partial regions in (c) and (e), respectively.

Table 7. The comparative results between the initial classification and the classification after optimization for three testing sets.

Sets	Before Optimization		After Optimization	
	Precision (%)	Recall (%)	Precision (%)	Recall (%)
S_1	87.4	79.1	87.3	81.1
S_2	92.9	77.5	93.3	82.3
S_3	80.8	77.6	81.5	80.2

Table 8. The confusion matrix among 9 object classes and the classification performance of each object class for initial classification on the testing dataset.

	Ground	Building	Fence	Telegraph Pole	Tree	Electric Wire	Street Light	Curb	Car	Recall (%)
Ground	1461348	628	14760	14	700	0	32	7276	182	98.4
Building	2834	720233	44230	710	335	1037	279	18	1474	93.4
Fence	6934	12217	346985	0	10872	24	0	432	0	91.9
Telegraph Pole	52	762	26	19923	603	0	1192	0	0	88.3
Tree	5063	13140	12904	441	537850	395	1458	3509	3059	93.1
Electric Wire	1070	391	4880	0	649	15066	98	0	0	68.0
Street Light	107	1943	136	1112	620	407	9116	0	0	67.8
Curb	38549	2181	5315	0	254	0	0	38643	707	45.1
Car	8157	1323	4273	46	1290	0	88	8983	35905	59.8
Precision (%)	95.9	95.7	80.0	89.6	97.2	89.0	74.3	65.7	86.9	
Overall precision: 86.0%, Overall recall: 78.4%.										

5 Discussion

It is challenging for comparing the classification results obtained in this paper with those of previous studies. At first, the quality and density of the experimental data are quite different. If the data quality is good, for example, the phenomenon of occlusion between objects is relatively small, or the data is more complete. If the Euclidean distance between objects belonging different classes is relatively large, it will help to obtain a higher overall classification accuracy. Secondly, except for the three-dimensional coordinates, in some previous algorithms, there are the reflection intensity, the RGB color values and other auxiliary information which can help improving the classification. In addition, different algorithms were tested with different kinds of objects and different numbers of object classes on different datasets, which will also affect the classification accuracy. For some object classes, such as trees, buildings and the ground, a high classification accuracy can be obtained. However, for some other less discriminatory object classes, such as buildings and fences, street lights and telegraph poles, it is difficult to efficiently classify them. In addition, the higher the number of object classes to be classified, the more difficult to obtain a high classification accuracy.

Despite the challenges aforementioned, here we aim to demonstrate that the performance of our proposed classification approach is comparable with those of previous algorithms. We only consider the point cloud classification studies of urban road environments obtained by using MLS. For the used nine object classes in this paper, we discuss the results reported in previous works for comparison with our results one by one in the following.

Table 9. The confusion matrix among 9 object classes and the classification performance of each object class for the optimized classification via graph cuts on the testing dataset.

	Ground	Building	Fence	Telegraph Pole	Tree	Electric Wire	Street Light	Curb	Car	Recall (%)
Ground	1473997	1138	15536	60	2269	48	212	7335	795	98.2
Building	6294	727605	43157	638	80	986	279	13	1351	93.2
Fence	3022	11077	355120	0	10617	24	0	448	0	93.4
Telegraph Pole	81	774	0	19850	632	0	1192	0	0	88.1
Tree	3439	7683	11731	586	537233	353	1521	3338	2782	94.5
Electric Wire	1012	150	3864	0	554	15159	18	0	294	72.0
Street Light	103	1340	109	1112	480	359	9005	0	0	72.0
Curb	31251	1832	1136	0	118	0	0	38926	499	52.8
Car	4915	1219	2856	0	1190	0	36	8801	35606	65.2
Precision (%)	96.7	96.7	81.9	89.2	97.1	89.5	73.4	66.1	86.2	
Overall precision: 86.3%, Overall recall: 81.0%.										

In previous works [17,18,25,41,46,56,57,64–68], the classification precisions of the ground are between 87% and 99% and the recall rates are between 60.6% and 99%. Most of them achieve a high precision of more than 94%, except for the precisions of 91.8% in [41] and 87.4% in [68]. The ground classification precision (96.7%) for our algorithm reaches an average level. The recall rate (98.2%) reaches a relatively high level which is only lower than ones in [57,65].

The classification of trees varied its precision from 60.1% to 99% and its recall rates from 51.1% to 99% in previous studies [25,41,45–47,57,64–66,68,69]. Our classification precision (97.1%) and recall rates (94.5%) of trees are at a moderate level. More than one species and varying sizes of trees will increase the tree classification difficulty which are existed in our testing data and in the previous studies [43,70]. However, in our work, we will get a superior tree classification results than the results in the previous studies [43,70]. This mainly results from the feature “percentage of scatter type points" which will effectively distinguish the trees and the other class.

For the building class, the precisions are between 82.2% and 99.28%, and the recall rates are between 86.7% and 95.7% in previous works [17,18,45,46,64,66–69]. Our results fall in these ranges. It can be seen from the confusion matrix that the buildings and fences are really easy to be misclassified. Because the buildings and fences are all the planes perpendicular to the ground surface, the distinction between these two classes is not very large except for the height-related features. This reason results in that the precision of the building class is not particularly high compared to the previous works that didn’t distinguish these two types of objects.

The precisions of cars in our results are higher than or approximately equal to those in previous works [17,18,46,67–69]. There are many parked cars at various orientations in our built dataset. Most of the parked cars were correctly classified. This demonstrates the rotation-invariant property of the proposed method as described in Section 3.2.1. However, the recall rates of cars are relatively lower than ones in previous works [17,18,46,67–69]. There are two possible reasons. At first, the data obtained by MLS is incomplete. The points of one side of the car are relatively dense, while there exist very few points in the other side due to acquisition occlusion, which can be obviously observed in Figure 3. This greatly increases the difficulty in identifying cars. In particular, curbs are similar to this kind of incomplete cars, making it easy for curbs to be classified as cars. Secondly, as cars are tightly attached to the ground, the segmentation will inevitably produce under-segmentation phenomenon,

which merge some of the points on the ground with the points on the car into one segment. Thus, these segments were often classified as cars, thus pulling down the recall rate of cars.

Both telegraph poles and electric wires usually appear in urban street scenes simultaneously. For these two object classes, our results are comparable with ones reported in [56,57,64,66]. Previous works report the precision rates of 34.65%-87% for telegraph poles and the precision rates of 9.34%-90% for electric wires. Most of them show a lower precision than our results. The recall rates of telegraph poles vary between 26% and 83.7%. As for electric wires, the recall rates vary between 13.4% and 87%. When comparing the precision and recall rate simultaneously, our algorithm shows excellent results. For example, Munoz *et al.* [56] achieved a higher telegraph poles precision (90.6%) and electric wires precision (90.6%) than ours for these two classes. But it generated a very low recall rate, especially for electric wires (13.4%). A high precision and low recall rate were achieved by Munoz *et al.* [57]. The recall rates of telegraph poles (82.07%) and electric wires (87%) reported in [64] are higher than or approximately equal to those in our results (88.1% and 72%). But it has a very low precision for these two classes (34.65% for telegraph poles and 9.34% for electric wires), which are far below the precisions obtained in this paper. The results in [66] are relatively balanced, but they are still lower than our results.

The results for fence classification are previously reported in [46,47]. Similar to ours, they suffered from the confusion between fences and buildings, which is a significant reason for the accuracy values being modest in both works.

To our knowledge, only Bremer *et al.* [65] had evaluated the classification precision of curbs. It obtains a high precision of 94% and a low recall rate of 69%, which are better than our results. The confusion between the ground and the curbs is the largest in [65]. However, their work didn't distinguish the car class, which has the largest confusion with curbs in our research.

In the previous works of the point cloud classification in the urban road environment, there is no direct classification of street lights. Therefore, we compared our results with those obtained from two object detectors described in [71,72]. Velizhev *et al.* [71] reported a precision of 72% and a recall rate of 82% for light poles. The precision is approximately equal to our result (73.4%), but the recall rate is higher than ours (72%). However, Velizhev *et al.* [71] only detected two object classes: light poles and cars. The difficulty of this classification is much lower than our study. The precision of light standards in [72] is 45% and the recall rate is 62%, which are lower compared with our results. Golovinskiy *et al.* [72] obtained the classification results of traffic lights and light standards at the same time. Similar to them, we distinguished the telegraph poles and street lights at the same time, which are easily confused with each other.

Overall, our proposed segmentation-based classification method can obtain better results than ones of the point-based classification methods [18,41,56,57,64,68] in the urban road environment. Compared with other segmentation-based classification methods [25,45–47,67], our method can generate a good classification result. Due to that the used point cloud dataset and the object classes to be classified are different, we cannot do a very fair comparison. But the proposed post-processing strategies and the graph-cuts-based classification optimization algorithm have been sufficiently proved to be efficient on our built dataset. It is believed that other existed algorithms can be improved by combining our post-processing and optimization techniques.

6 Conclusions

In this paper, we proposed an efficient three-stage classification framework for 3D point clouds in the urban road environment. At first, we applied a hierarchical and fast segmentation algorithm called P-Linkage to obtain initial segmentation results. In order to achieve a better segmentation result, we proposed a two-step post-processing strategy to merge the initial segments. Compared with the results of the initial segmentation, the segmentation after post-processing has obviously improved the classification accuracy and the recall rate. Secondly, we define a series of rational

features to distinguish the 9 different object classes and use the learned classifier to classify segments into these 9 classes. In the experiments, we used 3 kinds of classifiers (i.e. SVM, RF and ELM) and each classifier can achieve a good classification result. Thirdly, to reduce the misclassification of each object class and obtain a smoother classification results, we adopted graph cuts to optimally adjust the class labels of unreliable and small objects into the corresponding reliable objects. The experimental results on our built dataset show that the proposed optimization method can improve both the precision and recall rate of the initial point cloud classification, and achieve a superior result in three testing point cloud sets in the road environment.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (Project No. 41571436), the Hubei Province Science and Technology Support Program, China (Project No. 2015BAA027), the National Natural Science Foundation of China under Grant 91438203, LIESMARS Special Research Funding, and the South Wisdom Valley Innovative Research Team Program.

Bibliography

1. Scopigno, R.; Andujar, C.; Goesele, M.; Lensch, H.P.A. *3D Data Acquisition*; Eurographics Association, 2002.
2. KrÍźger, T.; Nowak, S.; Hecker, P. Towards autonomous navigation with unmanned ground vehicles using LiDAR. *Proceedings of the 2015 International Technical Meeting of the Institute of Navigation* **2015**.
3. Brenner, C. Extraction of features from mobile laser scanning data for future driver assistance systems. Advances in Giscience, Proceedings of the Agile Conference, Hannover, Germany, 2-5 June, 2009, pp. 25–42.
4. Selvam, S. Design and development of Integrated semi - autonomous fire fighting mobile robot **2015**. *4*.
5. Riveiro, B.; Diaz-Vilarino, L.; Conde-Carnero, B.; Soilan, M. Automatic segmentation and shape-based classification of retro-reflective traffic signs from mobile LiDAR data. *IEEE Journal of Selected Topics in Applied Earth Observations & Remote Sensing* **2015**, *9*, 1–9.
6. Yokoyama, H.; Date, H.; Kanai, S.; Takeda, H. Detection and classification of pole-like objects from mobile laser scanning data of urban environments. *International Journal of Cad/cam* **2013**, *13*.
7. Martin, R.; Pratihast, A.K.; Elberink, S.J.O. Tree modelling from mobile laser scanning data-sets. *Photogrammetric Record* **2011**, *26*, 361ÍC372.
8. Puttonen, E.; Jaakkola, A.; Litkey, P.; Hyypp?, J. Tree classification with fused mobile laser scanning and hyperspectral data. *Sensors* **2011**, *11*, 5158.
9. Shiravi, S.; Zhong, M.; Beykaei, S.A. Accuracy assessment of building extraction using LIDAR data for urban planning/transportation applications. The 2012 Conference of the Transportation Association of Canada Fredericton, New Brunswick, 2012.
10. Meyer, J.A.; Filliat, D. Map-based navigation in mobile robots: : II. A review of map-learning and path-planning strategies. *Cognitive Systems Research* **2015**, *4*, 283–317.
11. Roynard, X.; Deschaud, J.E.; Goulette, F. Fast and robust segmentation and classification for change detection in urban point clouds. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* **2016**, *XLI-B3*, 693–699.
12. Qin, R.; Gruen, A. 3D change detection at street level using mobile laser scanning point clouds and terrestrial images. *ISPRS Journal of Photogrammetry & Remote Sensing* **2014**, *90*, 23–35.
13. Aparajithan, S.; Shan, J. Urban DEM generation from raw LiDAR data. *Photogrammetric Engineering and Remote Sensing* **2005**, *71*, 217–226.
14. Guo, L.; Chehata, N.; Mallet, C.; Boukir, S. Relevance of airborne LiDAR and multispectral image data for urban scene classification using random forests. *ISPRS Journal of Photogrammetry and Remote Sensing* **2011**, *66*, 56–66.
15. Lalonde, J.F.; Vandapel, N.; Huber, D.F.; Hebert, M. Natural terrain classification using three dimensional ladar data for ground robot mobility. *Journal of Field Robotics* **2006**, *23*, 839ÍC861.

16. Behley, J.; Steinhage, V.; Cremers, A.B. Performance of histogram descriptors for the classification of 3D laser range data in urban environments. IEEE International Conference on Robotics and Automation, 2012, pp. 4391–4398.
17. Weinmann, M.; Urban, S.; Hinz, S.; Jutzi, B.; Mallet, C. Distinctive 2D and 3D features for automated large-scale scene analysis in urban areas. *Computers & Graphics* **2015**, *49*, 47–57.
18. Weinmann, M.; Jutzi, B.; Mallet, C. Semantic 3D scene interpretation: A framework combining optimal neighborhood size selection with relevant features **2014**. *ii-3*, 181–188.
19. Lee, I.; Schenk, T. Perceptual organization of 3D surface points. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* **2002**, *34*, 193–198.
20. Filin, S.; Pfelfer, N. Neighborhood systems for airborne laser data. *Photogrammetric Engineering & Remote Sensing* **2005**, *71*, 743–755.
21. Plazaleiva, V.; Gomezruiz, J.A.; Mandow, A.; GarcÍłacerezo, A. Voxel-based neighborhood for spatial shape pattern classification of LiDAR point clouds with supervised learning. *Sensors* **2017**, *17*.
22. Niemeyer, J.; Rottensteiner, F.; Soergel, U. Contextual classification of lidar data and building object detection in urban areas. *Isprs Journal of Photogrammetry & Remote Sensing* **2014**, *87*, 152–165.
23. Guo, B.; Huang, X.; Zhang, F.; Sohn, G. Classification of airborne laser scanning data using JointBoost. *Isprs Journal of Photogrammetry & Remote Sensing* **2015**, *100*, 71–83.
24. Blomley, R.; Jutzi, B.; Weinmann, M. Classification of airborne laser scanning data using geometric multi-scale features and different neighbourhood types. *Isprs Annals of Photogrammetry Remote Sensing & Spatial Informa* **2016**, *III-3*, 169–176.
25. Aijazi, A.K.; Checchin, P.; Trassoudaine, L. Segmentation based classification of 3D urban point clouds: A super-voxel based approach with evaluation. *Remote Sensing* **2013**, *5*, 1624–1650.
26. Wardale, J.; Mullen, L.; Howard, D. Object classification and recognition from mobile laser scanning point clouds in a road environment. *IEEE Transactions on Geoscience & Remote Sensing* **2015**, *54(2)*, 1–14.
27. Zhang, J.; Lin, X.; Ning, X. SVM-based classification of segmented airborne LiDAR point clouds in urban areas. *Remote Sensing* **2013**, *5*, 3749–3775.
28. Ni, H.; Lin, X.; Zhang, J. Classification of ALS point cloud with Improved point cloud segmentation and random forests **2017**.
29. Rutzinger, M.; H?fle, B.; Hollaus, M.; Pfeifer, N. Object-based point cloud analysis of full-waveform airborne laser scanning data for urban vegetation classification. *Sensors* **2008**, *8*, 4505–4528.
30. Krishnan, R. Object-oriented semantic labelling of spectralÍCspatial LiDAR point cloud for urban land cover classification and buildings detection. *Geocarto International* **2016**, *31*, 121–139.
31. Rollan, T.A.M.; Blanco, A.C. Assessment of point cloud analysis in improving object-based agricultural land cover classification using discrete lidar data in Cabadbaran, Agusan del Norte, Phillippines **2016**.
32. Rabbani, T.; Van Den Heuvel, F.; Vosselmann, G. Segmentation of point clouds using smoothness constraint. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* **2006**, *36*, 248–253.
33. Klasing, K.; Althoff, D.; Wollherr, D.; Buss, M. Comparison of surface normal estimation methods for range sensing applications. IEEE International Conference on Robotics and Automation, 2009.
34. Belton, D.; Lichti, D.D. Classification and Segmentation of Terrestrial Laser Scanner Point Clouds Using Local Variance Information. ISPRS Commission V Symposium 'Image Engineering and Vision Metrology', 2012.
35. Lu, X.; Yao, J.; Tu, J.; Li, K.; Li, L.; Liu, Y. Pairwise linkage for point cloud segmentation. XXIII ISPRS Congress, 2016.
36. Vosselman, G. Point cloud segmentation for urban scene classification. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* **2013**, *XL-7/W2(7)*, 257–262.
37. Golovinskiy, A.; Funkhouser, T. Min-cut based segmentation of point clouds. IEEE International Conference on Computer Vision Workshops (ICCV Workshops), 2009.
38. Strom, J.; Richardson, A.; Olson, E. Graph-based segmentation for colored 3D laser point clouds. IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010.
39. Anguelov, D.; Taskarf, B.; Chatalbashev, V.; Koller, D.; Gupta, D.; Heitz, G.; Ng, A. Discriminative learning of Markov random fields for segmentation of 3D scan data. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005.

40. Arya, S.; Mount, D.M.; Netanyahu, N.S.; Silverman, R.; Wu, A.Y. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM* **1998**, *45*, 891–923.
41. Yu, K.; Li, T.; Chen, J.; Wu, F.; Sun, C. Classification method for object feature extraction based on laser scanning data. *Communications in Computer and Information Science* **2013**, *398*, 155–165.
42. Bishop, C. *Pattern Recognition and Machine Learning (Information Science and Statistics)*; Springer, New York, 2007.
43. Lehtomäki, M.; Jaakkola, A.; Hyyppä, J.; Lampinen, J.; Kaartinen, H.; Kukko, A.; Puttonen, E.; Hyyppä, H. Object classification and recognition from mobile laser scanning point clouds in a road environment. *IEEE Transactions on Geoscience and Remote Sensing* **2016**, *54*, 1226–1239.
44. Lin, X.; Zhang, J.; Shen, J. Object-based classification of airborne LiDAR point clouds with multiple echoes. International Symposium on Image and Data Fusion, 2011, pp. 1–4.
45. Zhu, X.; Zhao, H.; Liu, Y.; Zhao, Y. Segmentation and classification of range image from an intelligent vehicle in urban environment. Ieee/rsj International Conference on Intelligent Robots and Systems, 2010, pp. 1457–1462.
46. Babahajiani, P.; Fan, L.; Gabbouj, M. Object recognition in 3D point cloud of urban street scene. Asian Conference on Computer Vision. Springer, 2014, pp. 177–190.
47. Zhou, Y.; Yu, Y.; Lu, G.; Du, S. Super-segments based classification of 3D urban street scenes. *International Journal of Advanced Robotic Systems* **2012**, *9*, 1.
48. Choe, Y.; Ahn, S.; Chung, M.J. Online urban object recognition in point clouds using consecutive point information for urban robotic missions. *Robotics and Autonomous Systems* **2014**, *62*, 1130–1152.
49. Zhang, J.; Lin, X.; Ning, X. SVM-based classification of segmented airborne LiDAR point clouds in urban areas. *Remote Sensing* **2013**, *5*, 3749–3775.
50. Tang, T.; Dai, L. Accuracy test of point-based and object-based urban building feature classification and extraction applying airborne LiDAR data. *Geocarto International* **2014**, *29*, 710–730.
51. Chehata, N.; Guo, L.; Mallet, C. Airborne LiDAR feature selection for urban classification using random forests. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 2009, Vol. 38, p. W8.
52. Mozos, O.M.; Stachniss, C.; Burgard, W. Supervised learning of places from range data using AdaBoost. IEEE International Conference on Robotics and Automation (ICRA), 2005.
53. Niemeyer, J.; Mallet, C.; Rottensteiner, F.; Soergel, U. Conditional random fields for the classification of LiDAR point clouds. ISPRS - International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 2011.
54. Niemeyer, J.; Rottensteiner, F.; Soergel, U. Contextual classification of point clouds using a two-stage crf. *Computer & Information Technology* **2015**, *XL-3/W2(3)*, 141–148.
55. Niemeyer, J.; Rottensteiner, F.; Soergel, U. Classification of urban LiDAR data using conditional random field and random forests. Joint Urban Remote Sensing Event (JURSE), 2013.
56. Munoz, D.; Vandapel, N.; Hebert, M. Directional associative markov network for 3-D point cloud classification. Fourth International Symposium on 3D Data Processing, Visualization and Transmission, 2008.
57. Munoz, D.; Bagnell, J.A.; Vandapel, N.; Hebert, M. Contextual classification with functional max-margin markov networks. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009.
58. Mount, D.M.; Arya, S. ANN: A Library for Approximate Nearest Neighbor Searching. https://www.cs.umd.edu/~mount/ANN/, 2010.
59. Nurunnabi, A.; West, G.; Belton, D. Outlier detection and robust normal-curvature estimation in mobile laser scanning 3D point cloud data. *Pattern Recognition* **2015**, *48*, 1404–1419.
60. Li, H.; Hu, W.; Yao, J. Anti-excessive filtering model based on sliding window. International Conference on Computer Science and Electronic Technology, 2015.
61. Chang, C.C.; Lin, C.J. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* **2011**, *2*, 27:1–27:27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.
62. Breiman, L. Random forests. *Machine Learning* **2001**, *45*, 5–32.
63. Huang, G.B.; Zhu, Q.Y.; Siew, C.K. Extreme learning machine: Theory and applications. *Neurocomputing* **2006**, *70*, 489–501.

64. Weinmann, M.; Jutzi, B.; Hinz, S.; Mallet, C. Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers. *ISPRS Journal of Photogrammetry and Remote Sensing* **2015**, *105*, 286–304.
65. Bremer, M.; Wichmann, V.; Rutzinger, M. Eigenvalue and graph-based object extraction from mobile laser scanning point clouds. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* **2013**, *5*, W2.
66. Huang, J.; You, S. Point cloud labeling using 3D convolutional neural network. Proc. of the International Conf. on Pattern Recognition (ICPR), 2016, Vol. 2.
67. Aijazi, A.K.; Serna, A.; Marcotegui, B.; Checchin, P.; Trassoudaine, L. Segmentation and classification of 3D urban point clouds: Comparison and combination of two approaches. Field and Service Robotics. Springer, 2016, pp. 201–216.
68. Choe, Y.; Shim, I.; Chung, M.J. Urban structure classification using the 3D normal distribution transform for practical robot applications. *Advanced Robotics* **2013**, *27*, 351–371.
69. Wang, Z.; Zhang, L.; Fang, T.; Mathiopoulos, P.T.; Tong, X.; Qu, H.; Xiao, Z.; Li, F.; Chen, D. A multiscale and hierarchical feature extraction method for terrestrial laser scanning point cloud classification. *IEEE Transactions on Geoscience and Remote Sensing* **2015**, *53*, 2409–2425.
70. Pu, S.; Rutzinger, M.; Vosselman, G.; Elberink, S.O. Recognizing basic structures from mobile laser scanning data for road inventory studies. *Isprs Journal of Photogrammetry & Remote Sensing* **2011**, *66*, S28–S39.
71. Velizhev, A.; Shapovalov, R.; Schindler, K. Implicit shape models for object detection in 3D point clouds. ISPRS Congress, 2012.
72. Golovinskiy, A.; Kim, V.G.; Funkhouser, T. Shape-based recognition of 3D point clouds in urban environments. IEEE International Conference on Computer Vision, 2009, pp. 2154–2161.