Classification of urban point clouds: A robust supervised approach with automatically generating training data

Zhuqiang Li, Liqiang Zhang, Ruofei Zhong, Tian Fang, Liang Zhang

***Abstract*—To reduce the cost of manually annotating training data for parsing outdoor scenes, we propose a supervised approach with automatically generating training data for classifying 3D point clouds of large-scale urban scenes. In this approach, the input point cloud is aggregated into point clusters, and the disjoint set union issue is combined with geometric attributes of each point cluster to obtain object segments. The prior knowledge among different classes is used to label the segments by using the decision-tree model. Then the initialized training samples are generated automatically. The confidence estimation for the labeling is employed to filter the mislabeled training samples. With the generated training data, we train a Random Forest classifier to create the initial classification of the 3D scene on the set of descriptors for each 3D point. The classification results are further optimized by Multi-Label Conditional Random Fields. Experimental results on five urban point clouds captured by different types of scanners (i.e. TLS, VLS and ALS datasets) demonstrate that the proposed approach achieves a competitive classification performance.**

***Index Terms*—Point cloud, classification, training data, prior knowledge.**

I. Introduction

The LiDAR technological evolution over the last few decades has provided high-resolution point clouds which open the doors to new application domains. LiDAR point clouds typically consist of complicated, densely aligned objects with large size variation. They allow the characterization of objects of interest in cities with unprecedented accuracy, and keep inventories up to date [1]. Availability of massive LiDAR scans data sets at the scale of entire cities has stimulated research on automated methods for urban object recognition. All of these factors, in conjunction to few labeled examples typically available, make urban point cloud classification automatically a very challenging problem.

In this paper, we propose a supervised approach with automatically generating training data for classifying 3D point clouds of large-scale urban scenes. Fig. 1 illustrates our proposed approach. Our approach consists of two main stages: the automatic generation of training data and the point cloud classification and optimization. In the first stage, an unsupervised algorithm combined with prior knowledge is applied to automatically generate training data from the input point cloud. First, we divide the point cloud into point clusters, and combine the disjoint set union with geometric attributes of each point cluster to obtain the segments, each of which contains more than one objects. The prior knowledge among different classes is described to label the segments by using the decision-tree model. Afterwards, the confidence estimation for the labeling is employed to filter the mislabeled training samples. In the second stage, different per-point descriptors have been proposed for TLS, VLS and ALS point clouds. The descriptor is composed of a set of light-weight 3D features. The random forest (RF) classifier is trained on the descriptor. With the automatically generated training data, the initial 3D classification of the urban environment is generated. The initial classification results are eventually optimized via the Graph-Cut framework.

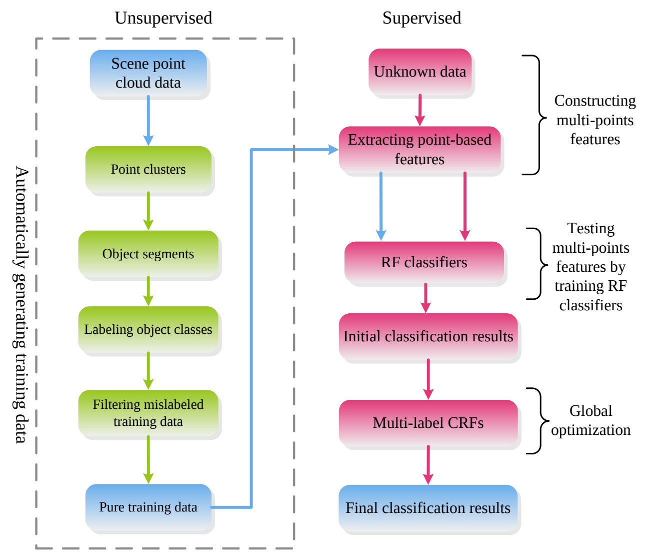

Unsupervised
Supervised
Scene point cloud data
Point clusters
Object segments
Labeling object classes
Filtering mislabeled training data
Pure training data
Automatically generating training data
Unknown data
Extracting point-based features
RF classifiers
Initial classification results
Multi-label CRFs
Final classification results
Constructing multi-points features
Testing multi-points features by training RF classifiers
Global optimization

Fig. 1. The flowchart of our proposed method.

It is worth mentioning that Zhang *et al.* [2] also presented a method for similar purpose. The method in [2] provide convincing parsing results of images and point cloud in street scenes but the differences between our work and theirs include that 1) Our approach can automatically extract terrain points,

This work was supported by the National Natural Science Foundation of China under Grant 41371324.

Zhuqiang Li, Liqiang Zhang, Ruofei Zhong, Liang Zhang are with Beijing Advanced Innovation Center for Imaging Technology, Capital Normal University, and the State Key Laboratory of Remote Sensing Science, Beijing Normal University, Beijing, China (e-mail: skybelongtous@foxmail.com, zhanglq@bnu.edu.cn, zrfsss@163.com).

Tian Fang is with Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong, China (e-mail: tianft@cse.ust.hk).

while [2] assumes relatively simple and flat terrains. Without accurate terrain extraction, classification of some objects is easy to fail. 2) Our approach can be adaptive to more widely point cloud datasets that may be highly noisy and heterogeneous. 3) Large-scale street scenes are parsed in [2] by jointing two acquisition modes, images and scanned point cloud, to infer the semantic category. The two modes have complementary characteristics. However, registration of the images and point cloud is a hard work. We only make use of the point cloud to recognize the objects in urban scenes, and achieve good classification accuracy even through a simple classifier.

The main contributions of our approach are threefold.

i) Propose an efficient and effective supervised approach for outdoor point cloud classification. We achieve comparative classification performance to that achieved by using manually labeled training data.

ii) An unsupervised algorithm combined with prior knowledge is applied to automatically generate training data from the input point cloud. The confidence estimation for the labeling is employed to filter the mislabeled training samples. To our best knowledge, this is the very first exploration of generating training data from the point clouds in urban environments without using other types of datasets.

iii) Construct a set of light-weight descriptors for each 3D point. We obtain the high classification accuracy of the 3D scene by employing a RF classifier on the descriptors for each 3D point and the multi-label Conditional Random Field (CRF).

II. RELATED WORK

The point cloud classification can be divided into unsupervised and supervised methods. In the unsupervised methods, Endre *et al*. [3] applied the Latent Dirichlet Allocation (LDA) to un-supervisely discover object classes in indoor scenes. Their method is designed for classifying objects which have been well segmented from the point clouds beforehand. In [4]-[8], some certain rules are presented to separate ground points from non-terrain points. Rutzinger *et al*. [9] developed an object-based method for classifying full waveform-driven LiDAR data into vegetation and non-vegetation objects. Lafarge and Mallet [10] distinguished four classes of interest, i.e. building, vegetation, ground and clutter, from 3D point clouds of urban environments by utilizing a Markov Random Field (MRF)-based optimization technique. In the similar framework, Gerke and Xiao [11] recognized four classes (buildings, trees, vegetated ground and sealed ground) by fusing airborne laser scanning (ALS) point clouds and images. In [12], during detecting and reconstructing complex 3-D watertight buildings, vegetation, buildings and terrains were accurately classified using the Graph-cut optimization technique followed by a hierarchical Euclidean clustering method. In order to classify urban 3D point clouds derived from oblique aerial imagery and vertical aerial imagery, a rule-based hierarchical semantic classification scheme has been developed [13]. The unsupervised classification strategy doesn't need training data and can work autonomously [11]. However, it is especially sensitive to the noise and clutter background. For the point clouds of large-scale urban environments, accurately classifying objects from them using the unsupervised approaches is a difficult task.

The supervised methods give a promising way to distinguish the interested objects in complex scenes containing many uncertainty and intricate relations among classes [14]. Mallet [15] used a point-based, multi-class support vector machine (SVM) to classify full-waveform LiDAR point clouds. In [14], a supervised classification method using the locally extracted features from a LiDAR point cloud was presented to identify on-ground objects. Based on an SVM classifier and evolutionary majority voting algorithm, Garc ía-Guti érrez *et al*. [16] developed a contextual classifier called SVM–EMV to model land use and land cover maps from high-resolution LiDAR data. By integrating a RF classifier into a CRF framework, Niemeyer [17] developed a context-based CRF classifier for urban LiDAR point clouds. Based on the construction of a minimum spanning forest from region markers, Tarabalka *et al*. [18] used spatial information described by a MRF to update the classification results produced by a probabilistic SVM. Negri *et al*. [19] developed a contextual classification method that used contextual information to displace the separation hyperplane obtained by the traditional SVM. In [20], neighboring points were clustered hierarchically to form a set of potential object locations. Then a graph-cut algorithm was used to segment the points surrounding those locations into foreground and background sets. Shape and contextual features were constructed for each point cluster. Finally, the SVM classifier was applied to label the objects into semantic groups. A multi-scale and hierarchical framework (MHPCs) was proposed in [21]. Based on the MHPCs, the features of point clusters are constructed by employing the LDA model. Later, they [22] combined the LDA with sparse coding to extract and encode the shape features of the multi-level point clusters. In [23], a patch-based framework that combines a 3-D patch-based match graph structure and a pairwise MRF model is applied to automatically label road scenes of colorized mobile LiDAR point clouds. This method can achieve a high performance for semantic labeling of road scenes.

The above supervised approaches have provided convincing classification results, but the training data is mostly generated by tedious and time-consuming manual labeling [2]. Moreover, the inherent computation consuming nature of classification further leads to difficulties in dealing with large-scale scenes. As a result, the classification performance achieved by using these approaches is typically compromised, reducing the practicality of the approaches.

III. AUTOMATIC GENERATION OF TRAINING DATA

Automatic generation of training data includes four steps: *i*) A clustering method is applied to aggregate the points in the point cloud into point clusters. *ii*) The geometric characteristics of each point cluster are combined to merge the point clusters into independent and complete segments. *iii*) Prior knowledge is utilized to label the segments for creating initial training data. *iv*) Mislabeled training samples are removed through RF cross-validation learning.

A. Generation of Point Clusters

The Euclidean clustering method [24] is employed to aggregate the input points into point clusters. In practice, each point cluster contains 40 points at least. It is noted that the point clouds of large-scale urban scenes typically exhibit significant missing data due to occlusion, as well as uneven point density. Therefore, the aggregation is often incomplete, and the points of an object can be divided into several point clusters. Thus, these point clusters should be merged into independent segments. Before the point clusters are merged, we compute the mean value of the distances between each point and its *k*-nearest neighbors for obtaining the approximate locations of the statistical outliers. Next, we integrate with the disjoint set union issue to merge the point clusters of planarity or non-planarity.

B. Merge Point Clusters to Object Segments

Through the above merging process, we have obtained a set of point clusters. Yet, for the point cloud with noise, outliers and areas of missing data, the segmentation is imperfect. To overcome the difficulty, we combine the disjoint set union issue [25] with geometric attributes of each point cluster to achieve object segments, each of which contains more than one objects.

B1) Geometric attributes

Four types of geometric attributes are employed for each point cluster:

$\mathbf{Adj}_{n\times n}$: The adjacency matrix is used to represent the spatial relationship between point clusters. We define d_{ij} is the distance between the *i*-th and *j*-th point clusters:

$$d_{ij} = \min_{p_k \in S_i, p_l \in S_j} \left\| p_k - p_l \right\| \tag{1}$$

where S_i and S_j are the *i*-th and *j*-th point clusters; p_k and p_l is the *k*-th and *l*-th points in S_i and S_j.

Suppose there are *n* point cluster sets. The average number of points in each set is *m*. Thus the time complexity for calculating d_{ij} is O(m^2). The time complexity for calculating the adjacency matrix is O($\frac{n(n-1)}{2} \cdot m^2$). To reduce the time complexity, we first find the neighboring point cluster sets $\{S_k, S_l \ldots\}$ of S_i through the kd-tree. Then the shortest distance of the two point clusters is converted into the shortest distance between two convex hulls. Eq. (2) is solved by using the quadratic programming solver for geometric optimization [26], which can greatly reduce the computational complexity.

$$\mathop{\arg\min}_{\substack{kdtree \\ S_k \in S_i}} \left| x^{\mathrm{T}} [p_i, p_j, \ldots -q_i, -q_j]^{\mathrm{T}}_{(p_i \in S_i, q_i \in S_k)} [p_i, p_j, \ldots -q_i, -q_j] x^{\mathrm{T}} \right| \tag{2}$$

$$\text{Subject to} \quad \begin{aligned} &\textstyle\sum_{i=1}^{r} x_i \\ &\textstyle\sum_{i=r+1}^{s} x_i = 1 \\ &x \geq 0 \end{aligned}$$

where points p_i and q_i are in the two point clusters S_i and S_k whose numbers of points are *r* and *s*, respectively. x^{T} is a vector which satisfies all constrains and makes the solution is bounded. Solving the quadratic program means we find such an optimal solution vector *x*.

$\mathbf{Nd}_{n\times 1}$: The average dispersion degree of the angles (/rad) between adjacent normal vectors in each point cluster. In the point cluster planarity merging process, point clusters often have good planarity, and their normals are consistent, but the structures of tree canopies are complex, thus they have large divergent angles between the normal vectors. The average dispersion degree of the *i*-th point cluster is computed using Eq. (3).

$$Nd_i = \frac{1}{r-1} \sum_{i=1}^{r-1} \left| n_{p_i} \cdot n_{p_i+1} \right| \tag{3}$$

where n_{pi} and n_{pi+1} are normal vectors of two adjacent points.

$\mathbf{Pl}_{n\times 1}$: The planarity is an important feature for the merging operations. We calculate the probability that the points in each point cluster are satisfied with the RANSAC fitting plane [27]:

$$Pl_i = \frac{1}{r} \sum_{j=0}^{r} \phi_j(p_j) \tag{4}$$

$$\phi_j(p_j) = \begin{cases} 1 & \text{if } \left| p_{xj} \cdot a + p_{yj} \cdot b + p_{zj} \cdot c \right| \cdot \left\| a^2 + b^2 + c^2 \right\|_2^{-1} \leq \varepsilon \\ 0 & \text{else} \end{cases} \tag{5}$$

where Pl_i describes the probability that the *i*-th point cluster is the planarity, and there are *r* points in this point cluster. $\phi_j(p_j)$ is data term of point p_j, which is used to determine whether p_j is a station point. *a*, *b* and *c* are the parameters of the RANSAC fitting plane. p_{xj}, p_{yj} and p_{zj} are the coordinates of the *j*-th point, respectively. ε is the tolerance parameter (in practice, ε=0.1m).

$\mathbf{H}_{\theta n\times 1}$: The angle between the dominant direction of the point cluster and the horizontal direction. It is a metric to determine the similarity of the two point clusters. In Eq. (6), $H_{\theta i}$ is the angle (/rad) between the *i*-th point cluster and the horizontal direction n_y.

$$H_{\theta_i} = \frac{1}{r} \sum_{j=0}^{r} \left| n_{p_j} \cdot n_y \right| \tag{6}$$

where n_{pj} is the normal vector of point *j*.

B2) Merge of point clusters

We use the disjoint set union issue [25] to generate complete independent point sets from the point clusters. Given a set with *n* elements U=S_1, …, S_n, and a set which has *m* relationships R=(i_1, j_1), (i_2, j_2), …, (i_m, j_m). *R* is true only if the following conditions are all satisfied:

i) For all S_a, $(S_a, S_a) \in R$ (Reflexive);

ii) Only if $(S_a, S_b) \in R$, $(S_b, S_a) \in R$ (Symmetry);

iii) If $(S_a, S_b) \in R$ and $(S_b, S_c) \in R$, $(S_a, S_c) \in R$ (Transitive).

The generated point clusters are independent equivalence classes. As they are merged, the geometric attributes are regarded as equivalence relations. The following two operations are performed. One is the **Find** operation which determines whether two equivalence classes are the same class. The other is the **Merge** operation which merges two equivalence classes into one class. Here we use the tree data structure to implement **Find** and **Merge** operations of the online equivalence class. Algorithm 1 and Algorithm 2 gives **Find** operation and **Merge** operation, respectively. Assign three domains, i.e. *root*, *parent* and *child*, to each element. *root*[*i*] is a bool type. If it is true, the current element is the root. *parent*[*i*] represents the parent node of the current element. If the current element is the root, *parent*[*i*] denotes the number of all its children, and the children are stored in the *child*[*i*]. *root*[*i*] is initialized to true, *parent*[*i*] is initialized to 1, *child*[*i*] is initialized to ∅.

Algorithm 1: Disjoint Set Union Issue: Find

```
Input: To find the element e. # ID of every point cluster
    #Assigned three domains for each point cluster, parent, root and child.
Output: The root node of e
 1: k ← e
 2: while root[k]==false do
 3:     k ← parent[k];
 4: end while
 5: # To reduce the search depth
 6: j ← e
 7: while j≠k do
 8:     pj←parent[j]
 9:     parent[j] ← k;
10:     j←pj;
11: end while
12: return j
```

Algorithm 2: Disjoint Set Union Issue: Merge

```
Input: To merge the element i, j. # ID of every point cluster
 1: $r_i$ ← i
 2: $r_j$ ← j
 3: if $r_i$==$r_j$ then
 4:     return; # i and j already belong to the same equivalence class, do not need to merge
 5: end if
 6: # According to the rule of the combined weight, the elements are merged into fewer elements and more nodes. Store their child nodes.
 7: if parent[$r_i$]> parent[$r_j$] then
 8:     parent[$r_i$] ←parent[$r_i$]+ parent[$r_j$];
 9:     root[$r_j$] ← false;
10:     parent[$r_j$] ←$r_i$;
11:     for all $child_j \in r_j$
12:         $child_i$=$child_i \cup child_j$;
13:     end for
14: else
15:     parent[$r_j$] ←parent[$r_j$]+ parent[$r_i$];
16:     root[$r_i$] ← false;
17:     parent[$r_i$] ←$r_j$;
18:     for all $child_i \in r_i$
19:         $child_j$=$child_j \cup child_i$;
20:     end for
21: end if
```

Next, we utilize the above obtained geometric attributes to merge point clusters with planarity. Algorithm 3 gives the algorithm of this process. For non-planar point clusters, we firstly calculate the distances from each point to its neighborhood. Suppose the distances confirm to the Gaussian distribution whose pattern is determined by the expectation and standard deviation of the distances. The points whose the mean distance is beyond the range of the standard deviation are regarded as sparse outliers. We make use of the adjacency matrix to merge the point clusters according to the distribution of outliers.

Algorithm 3: Merge point clusters into complete segment classes

```
Input: Adjacency matrix of all point clusters $\mathbf{Adj}_{n\times n}$; average normals dispersion $\mathbf{Nd}_{n\times 1}$; evaluation of planarity $\mathbf{Pl}_{n\times 1}$; the angle between the normal vector and the horizontality $\mathbf{H}_{\theta n\times 1}$.
 1: #Traverse the adjacency matrix $\mathbf{Adj}_{n\times n}$, with a smaller distance threshold value found in each point cluster of their adjacent point clusters $\mathbf{Adj}_{point\ cluster}$.
 2: for each point cluster $\mathbf{S_i}$ do
 3:     for each of $\mathbf{S_i}$ neighboring $\mathbf{S_j}$ do
 4:         if $\mathbf{Nd}_{[i]}$ >$\Delta\mu$ then
 5:             continue;
 6:         end if
 7:         if $\mathbf{Pl}_{n\times 1}$ >50% & $\mathbf{H}_{\theta[i]}$ and $\mathbf{H}_{\theta[j]}$ is similarity then
 8:             Merge($\mathbf{S_i}$, $\mathbf{S_j}$);
 9:         end if
10:     end for
11: end for
12: # For the non-planarity merge, we analyze the positions of the outlier points, through the distance $\mathbf{Adj}_{n\times n}$ consolidated way.
```

Fig. 2 illustrates the point clusters are merged into independent segments through the disjoint set union issue. In the scene, there are many different types of trees, and the tree points account for 81.9% of the total points. Most of building façades are occluded by the front trees. As a result, the obtained points on building facades typically exhibit significant missing data as well as uneven point density. The building façades and tree canopies are well merged as shown in Fig. 2(d).

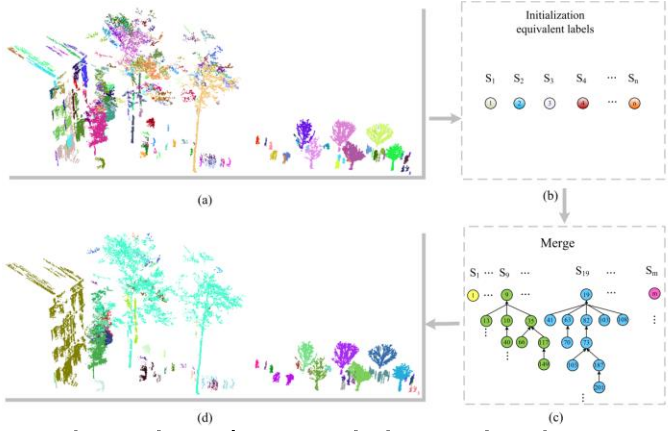

Initialization
equivalent labels
S_1 S_2 S_3 S_4 ... S_n
Merge
S_1 ... S_9 ... S_{19} ... S_m
(a)
(b)
(c)
(d)

Fig. 2. The point clusters of a TLS point cloud are merged into object segments. Different colors represent different cluster sets. (a) There are 222 point clusters generated by the Euclidean clustering algorithm. (b) Each point cluster is initialized to independent equivalence class labels. S_1~S_n represents n independent equivalence classes, where n is the number of the point clusters. (c) The tree structure of each segment after the merge operation. (d) The final merged result. 55 complete segments are generated.

C. Labeling of Each Class

Segments of different classes are extracted using the decision-tree model. The segmentation involves several discriminative properties of the classes:

i) Elevations of the segments: including the maximum height and the minimum local height of all points.

ii) The shape feature P_l of the planarity: We can separate the façades and roofs of buildings from trees with scattered surfaces. If $P_l > 0.8$, this segment is regarded to have a good planarity.

iii) The space projection: The ratio C_p of the canopy projection area to the diameter at breast height (DBH) area. For trees, $10 \le C_p \le 40$. For other three classes, the ratio C_p of the projected area to the bottom area of the object approximates to 1.

iv) The shape feature of the cylindrical models: For cars, we use the cylindrical model to determine their shape. The model parameters are derived by using the RANSAC. In general, the radius of a car cross-section is between 0.3m and1.0m. First, we compute the cylinder characteristic tolerance (M_{o1}), i.e. the number of the points in the cylindrical surface. Then, we compute the orientation angle M_{o2} according to the direction of the cylindrical axis ψ_{car} (Cy_x, Cy_y, Cy_z) and zenith direction $\boldsymbol{z}$ (0, 0, 1). The pedestrians and cars are well distinguished by determining whether they are satisfied with ($\psi_{car} \perp \boldsymbol{z}$), ($\psi_{pedestrian} // \boldsymbol{z}$).

The discriminative properties of the classes are taken as the weak prior knowledge to label the segments by using the decision-tree model. Then we achieve the initialized training samples.

D. *Filtering of Mislabeled Training Samples*

For the initialized training data generated through the weak priors knowledge, some of them would probably mislabeled. To remove the mislabeled samples in the training data, inspired by the method in [2], we remove mislabeled training samples through the confidence estimation.

D1) Weak priors features for confidence estimation

For each segment, we extract eight features *Φ* including the normal dispersion (D_d), planarity (P_l), direction angle of the major normal (H_θ), projection area ratio (C_p), the maximum and minimum local height ($height_{top}$, $height_{below}$), the cylinder characteristic tolerance (M_{o1}), and the angle between the cylinder axis and zenith direction (M_{o2}).

$$\Phi = \left[D_d \; P_l \; H_\theta \; C_p \; height_{top} \; height_{below} \; M_{o1} \; M_{o2} \right] \tag{7}$$

Since the dimensions of the different feature vectors are not the same, each dimension of all vectors in *Φ* needs to be normalized, and then we utilize the normalized *Φ* to train a binary RF classifier.

D2) Confidence estimation in the initial labeling

We compute the confidence probability of the initial labeled samples. The training samples with low confidence probability will be removed. The process for estimating confidence in the initial labeling is as follows.

i) The estimation process for each category follows the standard Leave-one-out cross-validation of multiple rounds with random data partitions. In each round of the cross-validation, the initial training data is randomly partitioned into training set and testing set.

ii) A binary RF classifier on the feature descriptor Φ is trained by the positive labeled samples from one category and the samples with the same number selected randomly from other classes as negative samples.

iii) Suppose the binary classifier performs better than random guess. If the initial labeling of a sample coming from the testing dataset is predicted to be positive, the probability that the initial labeling is correct is large than 50%. As the testing in each round of the cross-validation is based on random data partition and thus can be treated as independent testing, the more times the initial label of a sample agrees with the predicted label, the more probability its initial label is correct.

iv) Exchange the training dataset and testing dataset randomly, and repeat Steps *i*)-*iii*). We can get each object how many times are predicted positive.

v) According to [2], we use *P*(*L*, *n*) to denote the probability that a sample is classified as a positive sample *n* times during the N iterations, where $L \in \{+1, -1\}$ denotes the true label of the sample. Suppose the classification accuracy of the trained classifier is φ (φ>0.5), we take the iterations process as an approximate Binomial Distribution.

$$\mathrm{P}(n \mid L=+1)=C_N^n \varphi^n (1-\varphi)^{N-n} \tag{8}$$

$$\mathrm{P}(n \mid L=-1)=C_N^n (1-\varphi)^n \varphi^{N-n} \tag{9}$$

For a sample classified as a positive sample *n* times during the *N* iterations, the probability that its initial label is correct according to the Bayes' theorem is:

$$\begin{aligned}\mathrm{P}(L{=}{+}1 \mid n) &= \frac{\mathrm{P}(L=+1)\mathrm{P}(n \mid L=+1)}{\mathrm{P}(L=+1)\mathrm{P}(n \mid L=+1)+\mathrm{P}(L=-1)\mathrm{P}(n \mid L=-1)} \\ &= \frac{C_N^n \varphi^n (1-\varphi)^{N-n}}{C_N^n \varphi^n (1-\varphi)^{N-n} + \dfrac{\mathrm{P}(L=-1)}{\mathrm{P}(L=+1)} C_N^n (1-\varphi)^n \varphi^{N-n}}\end{aligned} \tag{10}$$

where P(*L*=-1) and P(*L*=+1) are unknown, the above step *iii)* we have made the assumption that the initial labeling is correct is large than 50%, so P(*L*=+1)≥P(*L*=-1). We can approximate the following:

$$\mathrm{P}(L{=}{+}1|n) \approx \frac{\varphi^n (1-\varphi)^{N-n}}{\varphi^n (1-\varphi)^{N-n} + (1-\varphi)^n \varphi^{N-n}} \tag{11}$$

where *N* =30, φ=0.6 in our experiment, we select the sample whose confidence estimation P(*L*=+1|*n*) ≥ 0.8 as the initial labeling is over correct.

Fig. 3 illustrates the segments of the point cloud is labeled by using prior knowledge. From this figure, we notice the four classes are separated precisely.

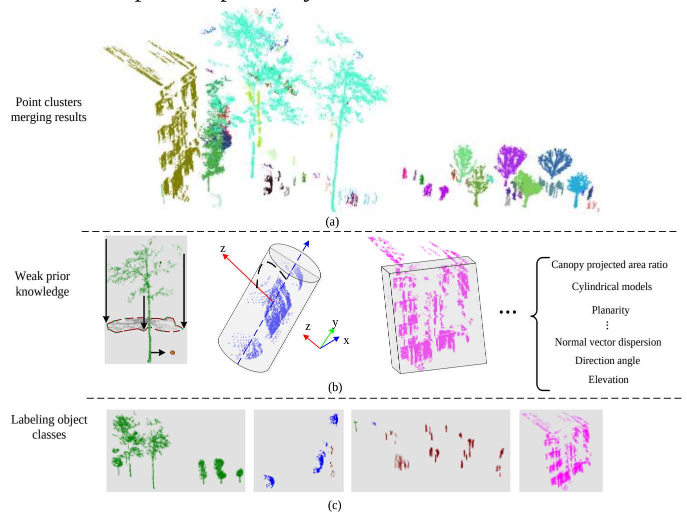


Fig. 3. Labeling of each class as the training data using weak prior knowledge. Different colors represent different objects. (a) The merged result. (b) Automatic extraction of four classes (buildings, trees, cars and pedestrians) from (a) based on weak prior knowledge. (c) The labeled four classes: trees, cars, pedestrians and buildings.

IV. Point Cloud Classification and Optimization

A. *Classification of point clouds*

In urban environments, the architectures of tree crowns are generally complicated, while shapes of building walls and roofs are smooth. The point-based features can preferably discriminate the shape-specific objects. For the local regions of the surfaces of tree trunks or cars, they usually have the planarity, and the points on them are often mis-classified into the building class through the point-based features. To overcome the drawback, the classification on the point clusters [21] or point clusters [29] have been developed. However, the shapes of the objects in a point cluster or point cluster are often incomplete because of over-segmentation or under-segmentation, which reduce the performance of the classification.

Since the point-based features are sensitive to noise, we use the spin images [30] of three scales (see Fig. 4) to capture the local feature operator. Let $\mathbf{N}_p=\{q|\ q$ is the neighboring points of p_i, and $|p_i-q|\leq r$ } be the point set within the sphere whose radius is r and the center is p_i. $\mathbf{N}_p$ is also taken as the support region of p_i. The normal vector of a point in $\mathbf{N_p}$ is taken as the rotational axis of the spin image. Eqs. (12) and (13) are applied to compute the coordinate of p_i in the spin image.

$$\alpha = \sqrt{\|q-p_i\|^2 - [n_i \cdot (q-p_i)]^2} \tag{12}$$

$$\beta = n_i \cdot (q-p_i) \tag{13}$$

where α denotes the x coordinate of p_i in the spin image; β denotes the y coordinate of p_i, and n_i denotes the normal vector of p_i.

The spin image can capture the majority of local shape information presented in a 3-D scene. We create a 6×6 bins spin image for each point. Next, the spin images are extracted from the three spheres for generating a 6×6×3 feature $\mathbf{F}_{\text{spin}}$. The dimension of $\mathbf{F}_{\text{spin}}$ is 108.

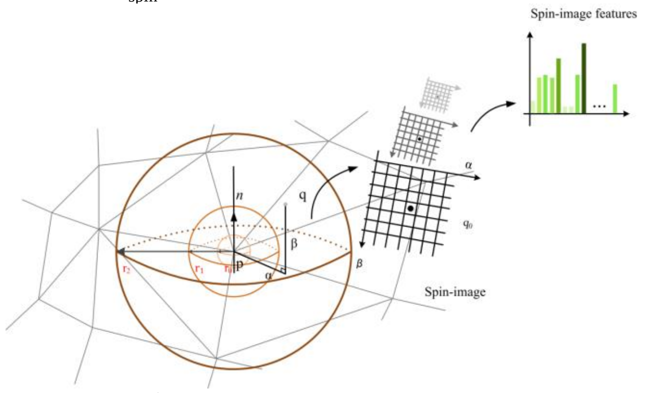


Fig. 4. Construction of the three-scale spin images.

Inspired by [31], we introduce the local feature descriptors and color texture feature. The local feature of $\mathbf{N}_p$ is expressed by the normal vectors n_i of each point p_i. Specially, we obtain n_i by calculating the eigenvalue and eigenvector of the covariance matrix of p_i. To emphasize, the covariance matrix C_i of each point p_i is derive from $\mathbf{N}_p$, and calculated by:

$$C_i = \frac{1}{g}\sum_{t=1}^{g}(p_t-\overline{p})(p_t-\overline{p})^{\mathrm{T}} \tag{14}$$

$$C_i \cdot \overrightarrow{v_j} = \eta_j \cdot \overrightarrow{v_j}, \quad j \in \{0,1,2\}$$

where T is the transposition operator, g is the number of the points in $\mathbf{N}_p$, $\overline{p}$ is the centroid of the point cluster in $\mathbf{N}_p$, and η_j is the jth eigenvalue of C_i, and $\overrightarrow{v_j}$ is the jth eigenvector of C_i. The eigenvalues η_0, η_1 and η_2 ($\eta_0>\eta_1>\eta_2$) are obtained by performing the Eigen decomposition of the covariance matrix C_i. The eigenvector $\overrightarrow{v_2}$ of the smallest eigenvalue η_2 in C_i is taken as the normal vector n_i of p_i.

In addition to the spin-image descriptor computed on three scales [32] and normal vector of each point, we also compute the following set of descriptors for each 3D P_i: mean RGB colors of the point as seen in the camera images; the LAB values of that mean RGB [33]; normal (n) at the 3D point; the point's height (*height*) above the estimated ground plane, and its "inverse height" (*height*$^{-1}$). For the TLS point cloud, the full descriptor per point p_i is:

$$\mathrm{F}_i^{TLS} = [RGB_i^{\mathrm{T}}\ LAB_i^{\mathrm{T}}\ n_i^{\mathrm{T}}\ \mathrm{F}_{spin}^{\mathrm{T}}\ height_i^{\mathrm{T}}\ height_i^{-1\mathrm{T}}] \tag{15}$$

F_i^{TLS} is a 119-dimensional vector.

For vehicle laser scanning (VLS) point cloud and airborne laser scanning (ALS) point clouds, we use the descriptors $\mathrm{F}_i^{\text{vls}}$ and $\mathrm{F}_i^{\text{als}}$ respectively in the following Eqs. (16) and (17):

$$\mathrm{F}_i^{\text{vls}} = [n_i^{\mathrm{T}}\ \mathrm{F}_{spin}^{\mathrm{T}}\ height_i^{\mathrm{T}}\ height_i^{-1\mathrm{T}}] \tag{16}$$

$$\mathrm{F}_i^{\text{als}} = [I_i\ Rnum_i\ Num_i\ n_i^{\mathrm{T}}\ \mathrm{F}_{spin}^{\text{als T}}\ height_i^{\mathrm{T}}\ height_i^{-1\mathrm{T}}] \tag{17}$$

where I_{i} is the refection intensity of the *i-th* point, $Rnum_i$ represents the pulse-echo order of the i-th point, and Num_i is the total number of echoes in the pulse. $\mathbf{F}_{\mathrm{i}}^{\text{vls}}$ is a 113-dimensional vector, and $\mathbf{F}_{\mathrm{i}}^{\text{als}}$ is a 116-dimensional vector.

We create the initial classification of the 3D scene by employing a RF classifier on the set of descriptors for each 3D point.

B. *Optimization of the Classification Results*

To reduce the misclassification caused by the above procedure, the multi-label CRF is adopted to globally optimize the classification results.

We define a graph $\mathcal{G}=(\boldsymbol{V}, \boldsymbol{\mathcal{E}})$ to organize the points in the point cloud, where $\boldsymbol{V}$ denotes the point nodes. $\boldsymbol{\mathcal{E}}$ is the set of the edges, which encodes the neighbouring point using the *KNN* algorithm. Assign a label l from the set of possible labels $\boldsymbol{L}=\{l_1, l_2,\cdots, l_n\}$ to each point p_i. The CRF is defined over this graph, and we aim to find the Maximum-A-Posteriori (MAP) labeling l^* of global points $\boldsymbol{V}$. It is equivalent to a Gibbs energy minimization problem of the general form:

$$l^* = \mathop{\arg\min}_{l\in\ L=\{l_1,l_2,\cdots,l_n\}} E(l) \tag{18}$$

where l_i is the class of the i-th point and the E(l) can be expanded into the following equation (19).

The initial classification result can be globally optimized through adding the probability of the classification result of each class as well as spatial consistency into the CRF optimization. We apply the efficient α-β swap algorithm [34], [35] to solve the multi-label energy objective function:

$$E(L\,|\,V) = \sum_{i\in V} D_i(l_i) + \lambda \cdot \sum_{(i,j)\in\varepsilon} V_{ij}(l_i,l_j) \tag{19}$$

$$D_i(l_i) = -\log_e \mathrm{P}(l_i\,|\,p_i) \tag{20}$$

where D_i is the data term; λ is the smooth coefficient of the propagation term. P ($l_i\,|\,p_i$) is the probability of label l_i. We wish the probability that the point i belongs to l_i is the maximum. In this case, the energy is the minimum. $V_{\mathrm{ij}}(l_i, l_j)$ is the propagation constraints of points i and j in a k-neighbourhood system. It enforces spatially smooth labeling solutions over the point by penalizing occurrences of its adjacent points i and j with different labels l_i and l_j ($l_i\neq l_j$).

$$V_{ij}(l_i,l_j) = \begin{cases} 0 & \text{if } l_i = l_j \\ w_{ij}\cdot 1 & \text{if } l_i \neq l_j \end{cases} \tag{21}$$

where the weight w_{ij} is defined as the angle cosine between the estimated normals of two points i and j.

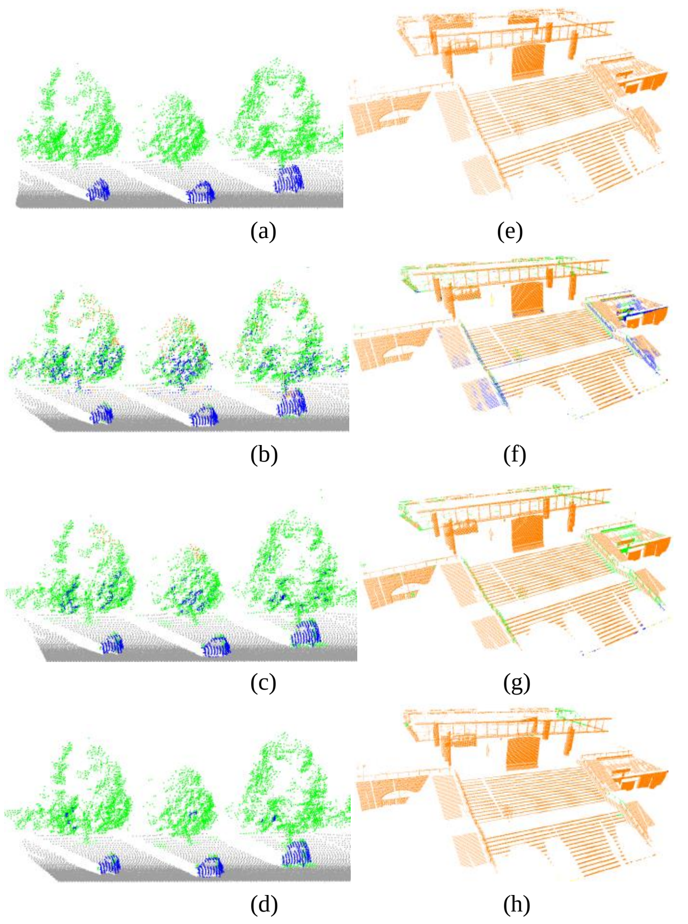


Fig. 5. Comparison of the supervised classification results obtained by using the RF classifier with the results optimized by performing the CRFs optimization.

We compare the classification result obtained by the above process. Fig. 5(a) and (e) show the ground truth of the street view and a complex building, respectively. Fig. 5(b) shows the classification result obtained using the linear SVM classifier. It is observed that there are many points on the foliage of the trees, which are wrongly classified as the car or building class. Fig. 5(c) illustrates the classification result of the RF classifier. Some wrongly classified points on the car by the SVM classifier have been recognized correctly. However, there are still some wrong points on the foliage. Fig. 5(d) shows the classification result which is optimized by the multi-label CRFs. Most of points belonging to other classes have been removed from the trees. As described in Fig. 5(f), the building classification result is obtained using the linear SVM classifier. Because the stair baluster is much sparse with foliage of tree on the structure, it is difficult to distinguish them. Fig. 5(g) illustrates the similar problem as Fig. 5(f). The final results are satisfactorily refined through the optimization procedure.

V. RESULTS AND DISCUSSIONS

A. *Experimental Datasets*

To validate the performance of our method, we perform both qualitative and quantitative evaluations on three different types of point clouds.

A1) TLS point clouds

The point clouds of three urban scenes were captured by TLS scanners in a single scan. The majority of objects that appear in the three urban scenes (Scenes I-III) are buildings, trees, cars and pedestrians. The obtained point clouds typically exhibit significant missing data due to occlusion, as well as uneven point density. In Scene I, there are a lot of trees with different shapes. The trees are self-occluded and at the same time they also occlude other objects such as buildings. In Scene II, there are many pedestrians moving on the roads. In Scene III, buildings with different shapes, e.g., the sloping stairs and low fences, were surrounded by trees and cars.

The terrain points in each of TLS point clouds are huge. To reduce computational cost, they are first segmented by using the mathematical morphological filtering algorithm in [8]. The number of the points in each of the classes is shown in Table 1.

TABLE 1. The number of the points in each class.

Scenes	Scene I	Scene II	Scene III
Car points	9,388	72,825,	171,220
Tree points	1,427,674	726,200	506,719
Pedestrian points	102,326	32,686,	65,372
Building points	204,096	182,483	496,307
Total points	1,743,434	1,015,204	1,239,618

Fig. 6 illustrates the optical images and the color point clouds mapping mosaic panoramic image of Scenes I, II and III.

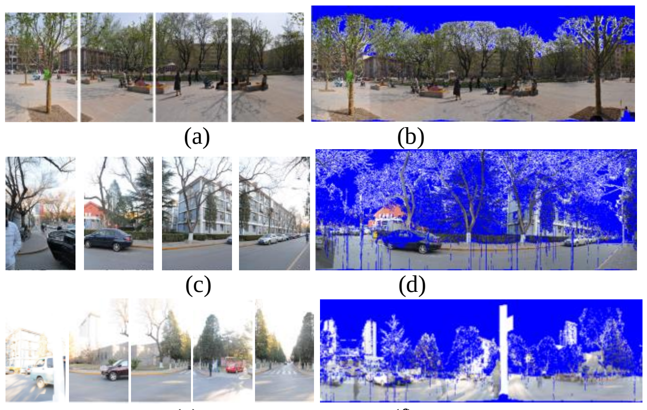


Fig. 6. The optical images and color point clouds mapping mosaic panoramic image. (a), (c) and (e) illustrate the optical images of Scenes I, II and III, respectively. (b), (d) and (f) illustrate the color point cloud mapping mosaic panoramic images of Scenes I, II and III, respectively. The blue color represents null values.

A2) Vehicle laser scanning data

Oakland 3-D Point Cloud Dataset [28] (Scene IV) was collected around CMU campus in Oakland, Pittsburgh, PA using Navlab11 equipped with side looking SICK LMS laser scanners. The majority of objects that appear in the street-view scene are buildings, trees, cars and ground. We use the label set: buildings, trees, cars, and ground provided by [28]. The statistics of the datasets are shown in Table 2.

A3) Airborne laser scanning data

The Vaihingen data [36] (Scene V) was covered by 10 strips acquired on 21 August 2008 by Leica Geosystems using a

Leica ALS50 system with 45° field of view and mean flying height 500m above the ground. The average strip overlap is 30%, and the median point density is 6.7 points/m^2. Point density varies considerably over the whole block depending on the overlap, but in the regions covered by only one strip, the mean point density is 4 points/ m^2. Multiple echoes and intensities were recorded. Due to the leave-on-conditions at the time of data acquisition, the number of points with multiple echoes is relatively low. The original point clouds were post-processed by strip adjustment to correct systematic errors in geo-referencing. Table 2 lists the information of the ALS points in Scene V. Fig. 7 illustrates an overview of the ALS data and the position of the test areas.

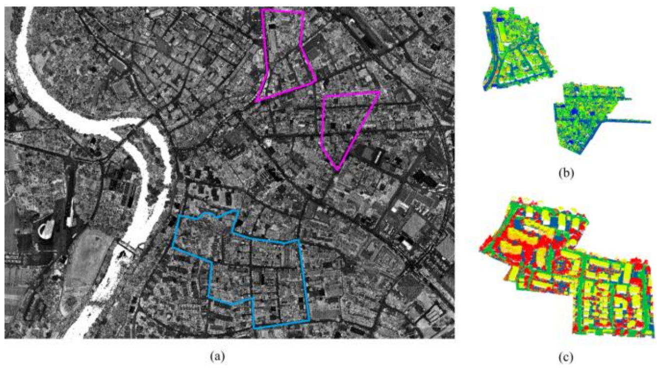


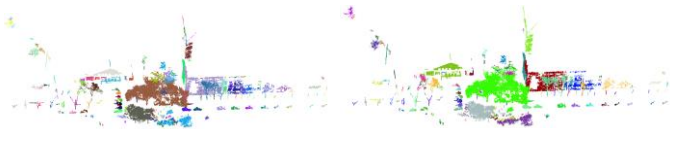
Fig. 7. Scene V: ALS data in Vaihingen city. (a) Intensity snapshot of the ALS data in Vaihingen city. The training/testing data and generalized data are in the blue border region and pink regions, respectively. (b) The generalized data without reference labels. (c) The ground truth data.

TABLE 2. The summary of Oakland (Scene IV) and Vaihingen (Scene V) datasets.

Scene IV	Cars points	Trees points	Ground points	Buildings points	Light poles points	Grass points	Shrubs points	Total points
	39,874	186,599	678,165	63,188	4,215	11,319	5,829	989,189
Scene V	4,614	182,778	193,723	191,365	-	-	-	572,480

B. *Evaluation of Automatically Generated Training Samples*

Fig. 8 illustrates the process for automatically generating training data and removing mislabeled training samples. As shown in Fig. 8(a), the points are aggregated into 260 point clusters. In Fig. 8(b), the point clusters that originally belong to the same object are merged into an object, and we obtain 182 segments. In Fig. 8(c), the objects in the red rectangles are the misclassified classes with the help of prior knowledge. Among them, the building fa çade points close to trees are divided into tree points, as well as the lower left corner of the car is mis-classified into low bushes. In Fig. 8(e), we remove the classes with low confidence probability based on the RF cross-validation learning. In practice, the probability $P(L=+1|n)$=85%. Fig. 8(d) and Fig. 8(f) are zoom-ins of the objects in the red line boxes of Fig. 8(c) and (e). It is noted that the wrongly classified objects are filtered out.



(a) (b)

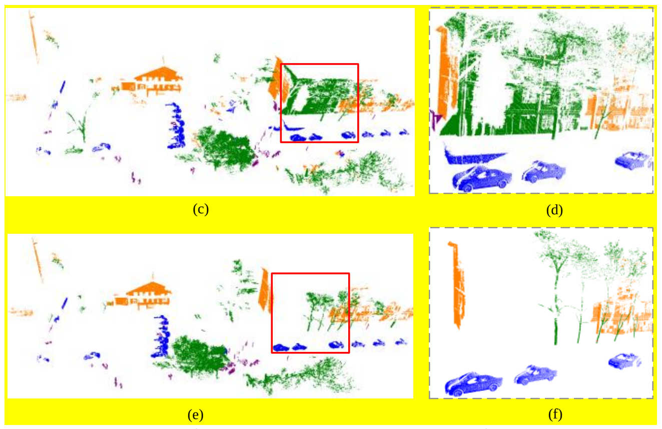


Fig. 8. The example of automatically generating and filtering mislabeled training data in Scene II. (a) Aggregation of points. Different colors represent different independent point clusters. The point cloud is divided into 260 point clusters. (b) The merged point clusters. (c) The automatic extraction of training data with weak prior knowledge, the points on pedestrians, trees, buildings, and cars are purple, green, orange and blue, respectively. (e) The pure training data after mislabeled samples are filtered. (d) and (f) illustrate zoom-ins of the objects in the red rectangles in (c) and (e).

Tables 3 list the quantitative results of the training data in Scenes I and II. From the tables, we notice that accuracies of the final generated training datasets are high.

TABLE 3. Accuracies of training data before/after mislabels are filtered in Scene I and Scene II.

	Scenes	CP/TEP	Accuracy	MF-CP/MF-TEP	Accuracy
Ground	Scene I	46,975/47,215	99.5%	46,975/47,215	99.5%
	Scene II	32,686/37,665	86.8%	32,641/33,110	98.6%
Trees	Scene I	572,541/572,541	100%	572,541/572,541	100%
	Scene II	228,131/301,567	75.6%	228,812/232,617	98.1%
Buildings	Scene I	32,552/34,388	94.7%	32,552/34,388	94.7%
	Scene II	80,358/87,415	91.9%	79,851/80,498	99.2%
Cars	Scene I	3,714/3,990	93.1%	3,714/3,880	95.7%
	Scene II	71,429/87,415	81.7%	71,429/73,110	97.7%

CP/TEP and *MF-CP/MF-TEP* represent the correct points/ the total extracted points before mislabels are filtered, and the correct points/ the total extracted points after mislabels are filtered, respectively.

As shown in Figs. 9-10, we use the dataset of Scenes IV-V to validate our method. Scene IV is that is a street-view. Trees are on both sides of the road. The low part of the buildings is missing due to occlusion, creating the effect of "floating in the air". At the labeling initialization, we mislabeled an incomplete building into a tree. The accuracy of trees is enhanced after wrong samples are filtered. Detailed results are shown in Table 4.

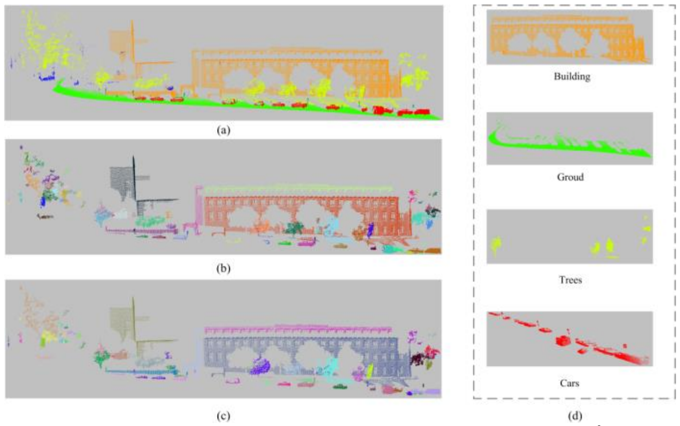


Fig. 9. Adaptive extraction of the training data in Scene IV (Slope: 4.7°). (a) The ground truth. Points on cars, trees, ground and buildings are colored in red, grass green, green and orange, respectively. (b) The segmentation result obtained using the Euclidean clustering algorithm. There are 100 point clusters. (c) Our merged result which has 73 complete segments. (d) The final obtained training samples after wrong samples are removed.

Table 4 lists the classification accuracy of the automatically generated training data from Scene IV. Since the data is a little sparse, tree structures are more dispersed. The labeling accuracy of the tree class is low because we mislabel an occluded fa çade into a tree. The accuracy is improved after the wrong samples are removed. The labeling accuracies of other classes are acceptable.

TABLE 4 Accuracy of training data before/after filtering mislabels in Scene IV.

	CP/TEP	Accuracy	MF-CP/MF-TEP	Accuracy
Ground	119,465/124,648	95.9%	11,9465/124,648	95.9%
Trees	7,337/10,914	67.2%	7,337/7,357	99.7%
Buildings	30,131/30,393	99.1%	30,131/30,393	99.1%
Cars	13,541/15,410	87.9%	12,695/13,541	93.8%

Similarly, from Fig. 10 and Table 5, it is noted that our method achieves good segmentation results of Scene V.

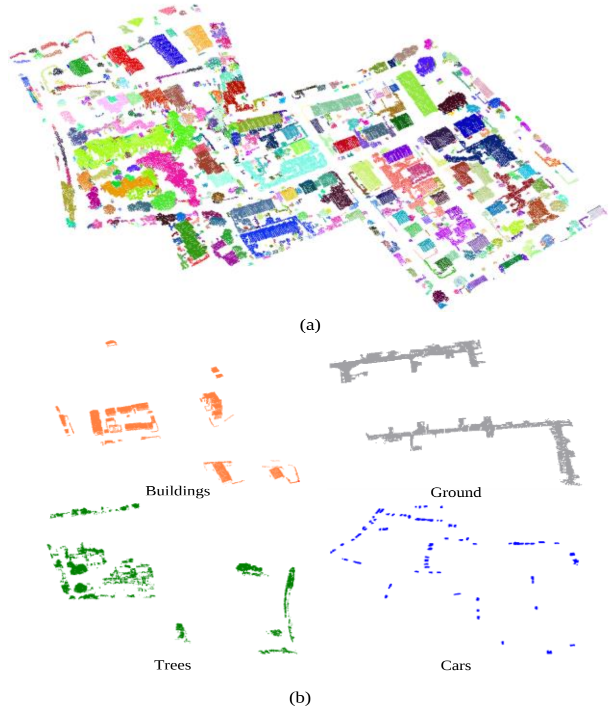


Fig.10. Adaptive extraction of the training data in Scene V (Slope: 3.5°). (a) Our merged result which has 1012 complete segments. (b) The final obtained training samples after wrong samples are removed.

TABLE 5. Accuracy of training data before/after filtering mislabels in Scene V.

	CP/TEP	Accuracy	MF-CP/MF-TEP	Accuracy
Ground	61,147/61,883	98.8%	61,147/61,883	98.8%
Trees	52,660/60,914	86.5%	49,059/53,117	92.4%
Buildings	30,979/34,193	90.6%	30,120/32,342	93.1%
Cars	2,618/3,410	76.8%	1,852/2,087	88.8%

From the above figures and tables, we conclude that the presented method can well distinguish the object classes from the point cloud of complex urban environments. The prior knowledge further helps us to correct some certain mislabeled classes and improves the labeling accuracy. The filtering scheme removes the mislabeled samples that are difficult to be recognized automatically from the generated training data. In this way, the final training data keeps a reliable confidence.

C. *Validation of Classification Performance*

The generated training datasets of the five point clouds are shown in Fig. 11. To demonstrate the classification performance of our method, we randomly select a part of training data. Table 6 lists the information of the final training data samples.

TABLE 6. The training samples selected from the generated training data.

Scenes	Total number of points	Ground points	Pedestrian points	Tree points	Building points	Car points
Scenes I-III	212,698	-	46,608	77,366	65,928	22,796
Scene IV	174,230	124,047	-	7,357	30,131	12,695
Scene V	149,429	61,883	-	53,117	32,342	2,087

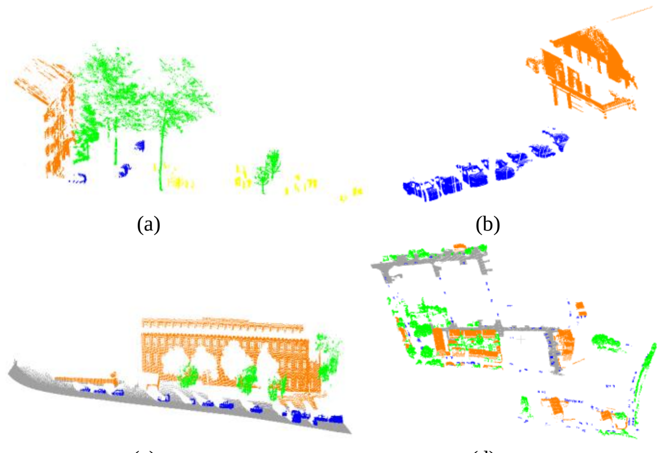
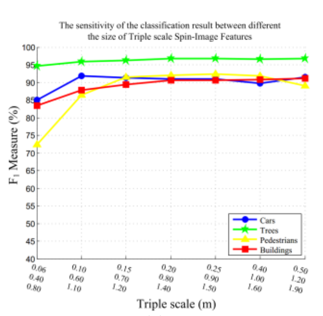


Fig. 11. Training data. In (a) and (b), a part of training data from Scenes I and II. Blue points are on cars, green points represent trees, yellow points are on pedestrians, and orange points are on buildings. In (c) and (d), a part of training dataset obtained from Scenes IV and V. Blue points are on cars, green points represent trees, gray points are on ground, and orange points are on buildings.

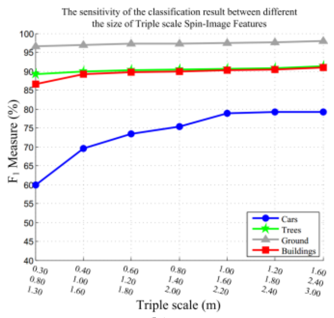
The following evaluations consist of two parts: *i*) Comparison of precision/recall of our method with those of other methods. *ii*) Scale selection of the spin image in the feature extraction, and the sensitivity of *k* in the *KNN* algorithm.

C1) Sensitivity of the parameters

In this sub-section, we analyze the sensitivity of the parameters in our method to the classification results in TLS and ALS datasets. The parameters include the spin-images with three scales, the number of the neighborhoods *k* in the KNN.



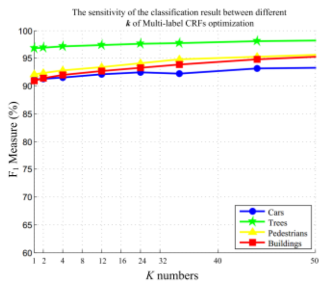
(a)



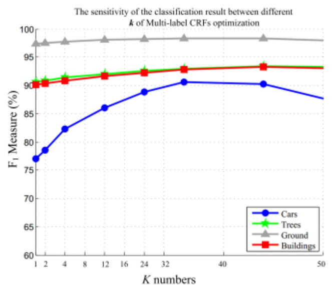
(b)

Fig. 12. The sensitivity of the radii that control the sizes of the three scales in the spin image extraction to the classification results (a) in Scene II and (b) Scene V, respectively.

Fig. 12 displays the sensitivity of the radii that control the sizes of the three scales in the spin images to the classification results (In the case, *k*=4). The vertical axis denotes F_1 measure, and the horizontal axis denotes the radii sizes of the three scales. Due to the small sizes and different distances from the scanners throughout the scanning, which make the point density inhomogeneous, the spin images have a difference in characterizing the shape information of the pedestrians and cars in a large scene. From Fig. 12(b), we observe that the scale of the feature extraction is larger than the TLS because of sparse density. In Fig. 12(a), the classification results of the car and pedestrian classes fluctuate greatly on range [0.06, 0.10], and those in Scene V range between [1.00, 1.60].



(a)



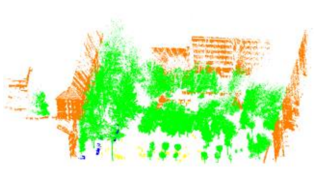
(b)

Fig. 13. The sensitivity of different *k* in the KNN for the multi-label CRFs optimization to the classification results (a) in Scene II and (b) Scene V, respectively.

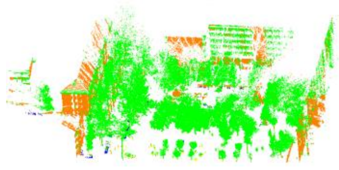
For the TLS and VLS datasets, the radii of the spheres that determine the scale size of the spin images are: r_0=0.2m, r_1=0.8m, r_2=1.4m. For the ALS datasets: r_0=0.8m, r_1=1.4m, r_2=2.0m. We test the sensitivity of different values of *k* in the *KNN* algorithm to the classification result. When *k*=12 in Fig. 13(a) and *k*=36 in Fig. 13(b), F_1 measures keep a high level. In general, the smaller the point density is, the larger the values of the radii and *k* are.

C2) Classification accuracy evaluation

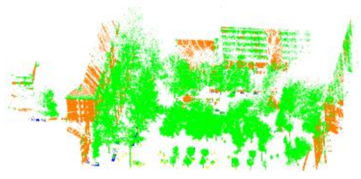
In the scenes I-IV, the radii of the three spheres used to compute the spin image features are r_0=0.2m, r_1=0.8m, r_2=1.4m; in Scene V, r_0=0.8m, r_1=1.4m, r_2=2.0m; *k*=4 in the *KNN* algorithm in the five scenes. Figs. 14-18 illustrate our classification results. The point-based features may cause misclassification in cluttered scenes. Especially for a few points on the corners of buildings and cars, they are difficult to be distinguished, e.g. Fig. 14(b), 15(b) and 16(b) show a small number of green points appeared on buildings and stairs, which means some points on buildings are misclassified as tree points. After the global optimization, the recognition quality of the points on the edges of cars and buildings are improved.



(a)

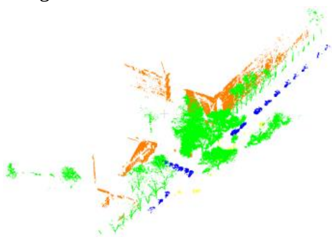


(b)

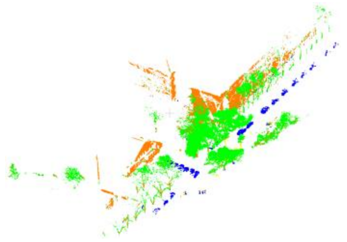


(c)

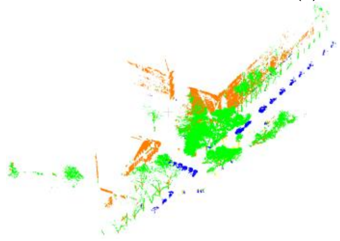
Fig. 14. The classification results in Scene I obtained using our approach. Points on cars, trees, pedestrians and buildings are colored in blue, green, yellow and orange, respectively. (a) Ground truth. (b)The classification result obtained using the RF classifier. (c) The classification result (b) optimized using the Multi-label CRFs.



(a)

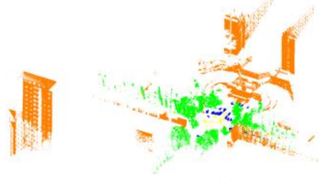


(b)

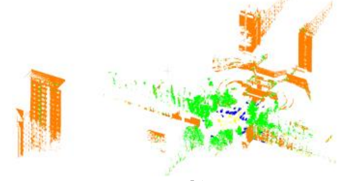


(c)

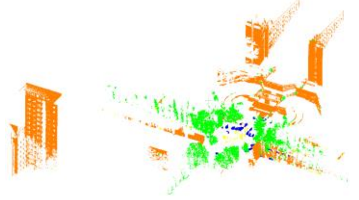
Fig. 15. The classification results in Scene II obtained through our approach. Points on cars, trees, pedestrians and buildings are colored in blue, green, yellow and orange, respectively. (a) Ground truth. (b)The classification result obtained using the RF classifier. (c) The classification result (b) optimized by the Multi-label CRFs.



(a)



(b)



(c)

Fig. 16. The classification results in Scene III obtained using our approach. Points on cars, trees, pedestrians and buildings are colored in blue, green, yellow and orange, respectively. (a) Ground truth. (b)The classification result

obtained using the RF classifier. (c) The classification result (b) optimized using the Multi-label CRFs.

From Scenes IV and V shown in Figs. 17-18, it is noted that the ground points have been distinguished from on-ground objects using our method, and the CRF-based optimization gets the maximum optimization of the class classification margin.

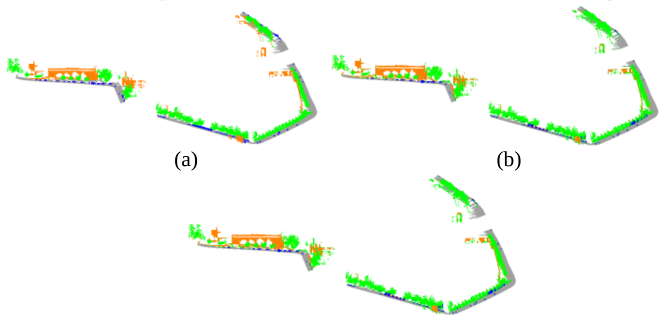


Fig. 17. The classification results in Scene IV obtained using our approach. Points on cars, trees, ground and buildings are colored in blue, green, gray and orange, respectively. (a) Ground truth. (b)The classification result obtained using the RF classifier. (c) The classification result (b) optimized using the Multi-label CRFs.

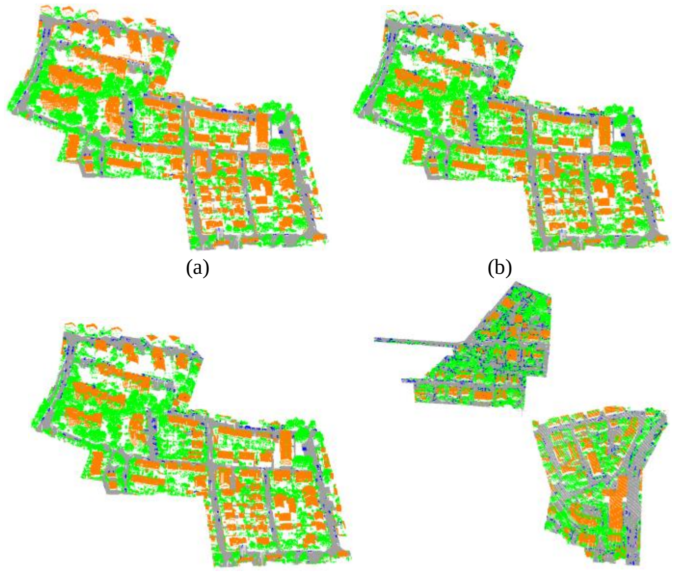


Fig. 18. The classification results in Scene V obtained by our approach. Points on cars, trees, ground and buildings are colored in blue, green, gray and orange, respectively. (a) Ground truth. (b)The classification result obtained by using the RF classifier. (c) The classification result (b) optimized by the Multi-label CRFs. (d)The classification result optimized by the Multi-label CRFs in generalized without reference-label data.

In Scene IV, there are many overlapped and closely neighboring objects. For example, trees overlap each other, and some buildings are close to trees. To demonstrate the effectiveness of our method applied to complicated urban scenes, we classify the point cloud in Scene IV into seven classes (Car, Tree, Ground, Building, Light pole, Grass, Shurb) and compare our calssification result with that of the method [21].

As shown in Fig. 19(e), the light poles are well seperated from the trees. In addition, our method has good local similarities in different classes in Fig. 19 (c), like trees and shurbs.

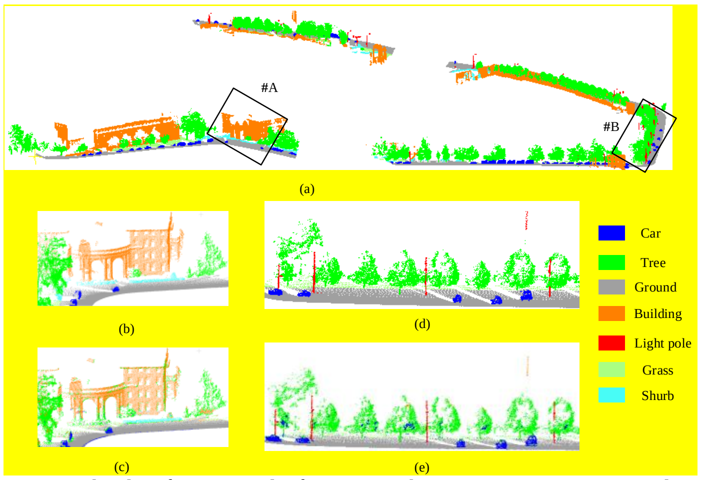


Fig. 19. The classification result of our approach. Points on cars, trees, ground, buildings, light poles, grass and shurb are colored in blue, green, gray orange, red, grass green and light blue, respectively. (a) Ground truth. (b) The close-up view of the ground truth in Area #A. (c) The close-up view of our result in #A. (d) The close-up view of the ground truth in Area #B. (e) The close-up view of our result in #B.

In Table 7, compared with our method, the precision and recall rate of the method [21] in terms of cars is less than 1.5% whereas cars are well recognized using our method as illustrated in Fig. 19(c).

TABLE 7. Precision/recall, F_1 measure and accuracy of the multi-class classification results in Scene IV.

Scene IV		Cars (%)	Trees (%)	Grou-nd (%)	Buildin-gs (%)	Light poles (%)	Gras-s (%)	Shru-bs (%)	*A* %
The method [21]	P	45.6	81.5	93.4	77.2	43.8	50.2	47.1	86.3
	R	66.2	79.6	92.9	81.3	31.9	53.4	39.8	
	F_1	54.0	80.5	93.1	79.2	36.9	51.7	43.1	
Our method	P	53.1	89.1	99.3	91.7	93.9	59.2	62.4	92.3
	R	85.3	90.7	96.4	91.6	56.3	67.6	45.9	
	F_1	65.5	89.9	97.8	91.7	70.4	68.9	52.9	

P, *R*, F_1 and *A* represent the precision, recall, F_1 measure and accuracy of the multi-class classification results, respectively.

Table 8 compares the classification performance of our method with those of the methods [21] and [23] in terms of precision/recall, F_1 measure and classification accuracy. From this table, we notice that the precisions and recalls of buildings and trees change quite stable, while those of cars and pedestrians fluctuate greatly. Simultaneously, we find that the number of points on cars in Scene I is less than 0.6% of tree points, which lead to a low classification precision due to the insufficient learning during the training stage. Through the global optimization, F_1 Measure reaches 60%. Similarly, in Scene V, the number of car points is less than 0.8% of all points, and the points on each car are few (about 15-130 points), so it is difficult to construct a discriminative feature to distinguish the cars from other classes. However, compared with the method [21], our method still has a better ability to deal with the sparse density and moving objects. In Scene V, shape and density of cars are similar to those of dwarf shrubs, it is easy to mis-classify the dwarf shrubs as cars, which leads to a low precision. F_1 Measure of cars reaches 75.4% by using the RF classifier. After the optimization, the precisions of all classes are improved significantly. Compared with the method [23], our method has a lower precision/recall in classifying cars and trees, but it has a higher classification accuracy of Scene IV.

TABLE 8. Precision/recall, F_1 measure and accuracy of the classification results.

Scene I	Cars (%)	Trees (%)	Pedestria -ns (%)	Buildin- gs (%)	Accuracy
The method [21]	52.9/45.4	95.4/98.3	68.5/62.7	86.9/83.6	92.4
	48.8	**96.8**	**65.5**	85.2	
Our Method	60.6/59.3	93.9/98.2	**86.4**/45.4	89.6/81.9	**93.0**
	60.0	96.0	59.5	**85.6**	
Scene II	Cars (%)	Trees (%)	Pedestrians (%)	Buildings (%)	Accuracy
The method [21]	77.4/85.6	98.2/95.6	84.9/69.4	83.7/92.3	93.3
	81.3	96.9	76.4	87.8	
Our Method	87.3/96.2	99.3/95.1	91.2/94.4	87.1/97.5	**95.6**
	91.5	**97.1**	**92.8**	**92.0**	
Scene III	Cars (%)	Trees (%)	Pedestrians (%)	Buildings (%)	Accuracy
The method [21]	83.7/86.4	95.9/90.1	78.8/77.5	89.0/93.3	90.2
	85.0	92.9	78.1	91.1	
Our Method	88.5/98.5	96.3/94.6	85.7/99.9	97.9/93.8	**95.1**
	93.3	**95.5**	**92.3**	**95.9**	
Scene IV	Cars (%)	Trees (%)	Ground (%)	Buildings (%)	Accuracy
The method [21]	49.6/66.8	83.6/84.2	93.4/92.5	80.9/76.1	86.7
	56.9	83.8	92.9	78.4	
The method [23]	99.7/99.5	96.3/92.4	95.0/95.9	76.8/88.2	92.9
	99.6	**94.3**	95.5	82.1	
Our Method	53.6/77.2	87.5/86.3	99.9/97.1	78.9/89.5	**93.4**
	63.3	86.9	**98.4**	**83.9**	
Scene V	Cars (%)	Trees (%)	Ground (%)	Buildings (%)	Accuracy
The method [21]	60.5/73.9	84.3/94.6	95.1/90.4	93.6/83.7	90.2
	66.5	89.2	92.7	88.4	
Our Method	70.1/99.8	85.7/97.9	98.0/97.4	98.3/84.5	**93.3**
	82.3	**91.4**	**97.7**	**90.9**	

Our method was implemented on the computer with the processor of Intel Core i7-4790K 3.60GHz and 8G RAM. During the whole classification procedure, the feature extraction occupies most of computational time, nearly 78% of the whole procedure. Comparison of the computional time is listed in Table 9. From this table, we notice our method has a higher efficiency than the method [21].

TABLE 9. Comparison of time costs in Scenes I and IV.

Methods	Scenes	MPFs extraction (min)	RF learning (m)	RF testing (m)	CRFs Time (s)	All Time (min)
Our method	I	88.3	4.6	18.3	61.9	***112.2***
	IV	49.6	3.5	8.6	41.8	***62.3***
		MHPCs extraction (min)	learn LDA (m)	AdaBoost learning time(m)	AdaBoost testing (m)	All Time (min)
The method [21]	I	52.4	20.1	20.7	22.6	115.8
	IV	38.5	15.3	14.6	18.8	87.2

VI. CONCLUSIONS

We have presented an automatic method for outdoor point cloud classification. The key feature of the presented method is its ability to automatically and efficiently generate training data from the input dataset without manual labeling. Moreover, the approach can be widely applicable to various types of urban point clouds, because it makes few assumptions on the properties of the datasets. Compared with previous methods, the presented method has achieved impressive performance in terms of higher classification accuracy and lower computation cost.

The limitations of the presented method lie in: *i*) If the points on the objects are quite few due to their small sizes or collusions, it is difficult to automatically generate the training data for these classes. *ii*) The employed features mainly describe the shape of the objects, thus different classes with similar shapes are often mis-classified into one class. For example, the dwarf shrubs and cars in Scene V shown in Fig. 18 are often mis-classified into dwarf shrubs. A possible way to partially alleviate the mistake is to rely on contextual features. For example, cars are found on streets, often in a line, whereas lampposts are found on sidewalks, sometimes in a pattern [19].

In the future, we plan to make use of the joint optimization of feature learning and classifier learning in a coherent way to enhance the recognition performance of complex large-scale outdoor environments.

REFERENCES

[1] G. Camps-Valls, D. Tuia, L. Bruzzone, and J. A. Benediktsson, "Advances in hyperspectral image classification," *IEEE Signal Proc. Mag.*, vol. 31, no. 1, pp. 45–54, Jan. 2014.

[2] H. Zhang, J. Wang, T. Fang and L. Quan, "Joint Segmentation of Images and Scanned Point Cloud in Large-Scale Street Scenes with Low-Annotation Cost," IEEE Trans. Image Process., vol. 23, no. 11, pp. 4763-4772, Nov. 2014.

[3] F. Endres, C. Plagemann, C. Stachniss, and W. Burgard, "Unsupervised discovery of object classes from range data using latent Dirichlet allocation," in *Proc. Robotics: Science and Systems*, Washington, 2009.

[4] K. Kraus, N. Pfeifer, "Determination of terrain models in wooded areas with airborne laser scanner data," *ISPRS J. Photogramm. Remote Sens*., vol. 53, no. 4, 193-203, 1998.

[5] G. Vosselman, "Slope based filtering of laser altimetry data," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, 33 (Part B3), 935–942, 2000.

[6] D. Mongus, B. Žalik, "Parameter-free ground filtering of LiDAR data for automatic DTM generation," *ISPRS J. Photogramm. Remote Sens.*, vol. 67, 1–12, 2012.

[7] Z. Yu, C. XU, J. Liu, O. C. Au, and X. Tang, "Automatic object segmentation from large scale 3D urban point clouds through manifold embedded mode seeking," In *Proc. 19th ACM international conference on Multimedia*, 2011.

[8] D. Chen, L. Zhang, Z. Wang, and H. Deng, "A mathematical morphology-based multi-level filter of LiDAR data for generating DTMs," *Science China Information Sciences*, vol. 56, no. 10, pp. 1-14, Oct. 2013.

[9] M. Rutzinger, B. Höfle, M. Hollaus, and N. Pfeifer, "Object-based point cloud analysis of full-waveform airborne laser scanning data for urban vegetation classification," *Sensors*, vol. 8, no. 8, pp. 4505–4528, 2008.

[10] F. Lafarge and C. Mallet, "Creating large-scale city models from 3d-point clouds: A robust approach with hybrid representation," *Int. J. Comput. Vis.*, vol. 99, no. 1, pp. 69–85, Feb. 2012.

[11] M. Gerke, J. Xiao, "Fusion of airborne laserscanning point clouds and images for supervised and unsupervised scene classification," *ISPRS J. Photogramm. Remote Sens*., vol. 87, pp. 78–92, Jan. 2014.

[12] S. H. Sun and C. Salvaggio, "Aerial 3D building detection and modeling

from airborne LiDAR point clouds," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 6, no. 3, pp. 1440–1449, Jun. 2013.

[13] J. Rau, J. Jhan, and Y. Hsu, "Analysis of Oblique Aerial Images for Land Cover and Point Cloud Classification in an Urban Environment," *IEEE Trans. Geosci. Remote Sens*., vol. 53, no. 3, pp.1304-1319, Mar. 2015.

[14] B. Guo, X. Huang, F. Zhang, and G. Sohn, "Classification of airborne laser scanning data using JointBoost," *ISPRS J. Photogramm. Remote Sens*., vol. 100, pp. 71-83, Feb. 2015.

[15] C. Mallet, "Analysis of Full-Waveform Lidar Data for Urban Area Mapping," Ph.D. thesis, Télécom ParisTech, 2010.

[16] J. García-Gutiérrez, D. Mateos-García, M. Garcia, and J. Riquelme-Santos, "An evolutionary-weighted majority voting and support vector machines applied to contextual classification of LiDAR and imagery data fusion," *Neurocomput*., vol.163, no. 2, pp. 17–24, Sep. 2015.

[17] J. Niemeyer, F. Rottensteiner, and U. Soergel, "Contextual classification of LiDAR data and building object detection in urban areas," *ISPRS J. Photogramm. Remote Sens*., vol. 87, pp. 152–165, Jan. 2014.

[18] Y. Tarabalka, J. Chanussot, and J.A. Benediktsson, "Segmentation and classification of hyperspectral images using minimum spanning forest grown from automatically selected markers," *IEEE Trans. Syst. Man Cybern. B Cybern*., vol. 40, no. 5, pp. 1267–1279, Oct. 2010.

[19] R. G. Negri, L. V. Dutra, and S. J. S. Sant'Anna, "An innovative support vector machine based method for contextual image classification," I*SPRS J. Photogramm. Remote Sens*., vol. 87, pp. 241–248, Jan. 2014.

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

[21] Z. Wang, L. Zhang, T. Fang, T. Mathiopoulos, X. Tong, H. Qu, Z. Xiao, F. Li, and D. Chen, "A Multiscale and Hierarchical Feature Extraction Method for Terrestrial Laser Scanning Point Cloud Classification," *IEEE Trans. Geosci. Remote Sens*., vol. 53, no. 5, pp. 2409-2425, May 2015.

[22] Z. Zhang, L. Zhang, X. Tong, Z. Wang, B. Guo, X. Huang, Y. Wang, "A Multi-Level Point Cluster-based Discriminative Feature for ALS Point Cloud Classification," *IEEE Trans. Geosci. Remote Sens*., vol. 54, no. 6, 3309-3321, Jun. 2016.

[23] H. Luo, C. Wang, C. Wen, Z. Cai, Z. Chen, H. Wang, Y. Yu, and J. Li, "Patch-based semantic labeling of road scene using colorized mobile LiDAR point clouds," *IEEE Transactions on Intelligent Transportation Systems*., vol. 17, no. 5, pp. 1286 – 1297. May 2016.

[24] R. B. Rusu, "Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments," Ph.D thesis, Computer Science department, Technische Universitaet Muenchen, Germany, Oct. 2009.

[25] Z. Yu, Y. Yu, G. Lu, and S. Du. "Super-segments based classification of 3d urban street scenes," *Int J. Adv. Robot. Syst.*, vol. 9, no. 248, 2012.

[26] B. Gärtner, and S. Schönherr. "An efficient, exact, and generic quadratic programming solver for geometric optimization," in *Proc. 16th ACM Symposium on Computational Geometry*, 2000.

[27] M . A. Fischler and and R. C Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *ACM Commun*., vol. 24, no. 6, pp. 381–395, Jun. 1981.

[28] D. Munoz, J.A. Bagnell, N. Vandapel, and M. Hebert, "Contextual classification with functional max-margin markov networks," in *Proc. IEEE Int. Conf. Comput. Vis*., Miami, FL, USA, pp. 975-982, Jun. 2009.

[29] E. H. Lim and D. Suter, "3D terrestrial LIDAR classifications with point clusters and multi-scale Conditional Random Fields," *Comput. Aided Design*, vol. 41, no. 10, pp. 701-710, Oct. 2009.

[30] M. Lehtomäki, A. Jaakkola, J. Hyyppä, J. Lampinen, H. Kaartinen, A. Kukko, E. Puttonen, and H. Hyyppä, "Object Classification and Recognition From Mobile Laser Scanning Point Clouds in a Road Environment," *IEEE Trans. Geosci. Remote Sens*., vol. 54, no. 2, pp. 1226 – 1239, Feb. 2016.

[31] A. Martinovic, J. Knopp, H. Riemenschneider, and L. Van Gool, "3D All The Way: Semantic Segmentation of Urban Scenes From Start to End in 3D," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*., 2015.

[32] Z. Li, L. Zhang, X. Tong, B. Du, Y. Wang, L. Zhang and Z. Zhang, "A Three -step Approach for TLS Point Cloud Classification," *IEEE Trans. Geosci. Remote Sens*., vol. 54, no. 9, pp. 5412-5424, Sep. 2016.

[33] H. Riemenschneider, A. Bodis-Szomoru, J. Weissenberg, and L. Van Gool. "Learning Where To Classify In Multi-View Semantic Segmentation," in *Proc. Eur. Conf. Comput. Vis*., Zurich, Switzerland, Sep. 2014, pp. 516-532.

[34] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Trans. Pattern Anal. Mach. Intell*., vol. 26, no. 9, pp. 124–1137, 2004.

[35] B. Fulkerson, A. Vedaldi, and S. Soatto, "Class segmentation and object localization with superpixel neighborhoods," in *Proc. 12th IEEE Int. Conf. Comput. Vis*., 2009, pp.670-677.

[36] N. Haala, H. Hastedt, K. Wolf, C. Ressl, and S. Baltrusch, "Digital photogrammetric camera evaluation–generation of digital elevation models," *Photogrammetrie-Fernerkundung-Geoinformation*, vol. 2, 2010, pp. 99-115.