Pole-Like Object Detection and Classification from Urban Point Clouds

Jing Huang[1] and Suya You[2]

Abstract— **This paper focuses on detecting and classifying pole-like objects from point clouds obtained in urban areas. To achieve our goal, we propose a system consisting of three stages: localization, segmentation and classification. The localization algorithm based on slicing, clustering, pole seed generation and bucket augmentation takes advantage of the unique characteristics of pole-like objects and avoids heavy computation on the feature of every point in traditional methods. Then, the bucket-shaped neighborhood of the segments is integrated and trimmed with region growing algorithms, reducing the noises within candidate's neighborhood. Finally, we introduce a representation of six attributes based on the height and five point classes closely related to the pole categories and apply SVM to classify the candidate objects into 4 categories, including 3 pole categories light, utility pole and sign, and the non-pole category. The performance of our method is demonstrated through comparison with previous works on a large-scale urban dataset.**

I. Introduction

There are several categories of objects in a typical urban scene, including buildings, cars, trees and poles. In this paper, we focus on detection of pole-like objects, including utility poles, street lights, traffic lights, road signs, flag poles and parking meters. With the correctly identified instances, there can be many potential applications such as navigation for robots, autonomous driving and urban modeling.

Early works use images or videos to detect pole-like objects [1]. As 3D data become popular nowadays, the advantage that 3D data can avoid problems such as illumination and background confusion in 2D has been realized. On the other hand, the fact that 3D sensed datasets contain a large number of points calls for the efficiency of algorithms. While we can classify and extract all linear clusters from arbitrary directions (see Sec. VI-A), most pole-like objects are in the upright direction, even if the terrain is steep, due to safety and usability requirements. In case the data come without a regular well-aligned coordinate system, we can either first roughly fit the ground and get the upright direction, or manually select the z-direction. After that, we are able to apply a fast vertical bounding-box-based method to extract all possible locations of pole-like objects.

On the other hand, simple clustering methods based on spatial proximity are not feasible since the ground always connect everything together. Also, in large-scale datasets such as urban scenes, the majority of data belong to large scale planes including ground and building facade, which is not the focus of pole detection. Therefore, most previous works tend to use a plane-fitting technique to remove the ground. If the ground is removed perfectly, then all the objects over it seem to be properly segmented. Unfortunately, the ground is not always planar, meaning that a brute-force fitting may fail. Another approach is to classify the local shapes that are planar, which yields accurate classification of the points. However, this method costs too much time on computing features for every point. In contrast, our method uses the simple horizontal slicing to separate the ground and then applies clustering based on Euclidean distance on each layer, avoiding plane fitting and computation of features on every point. The problem left is to reassemble the broken parts while avoiding the interference of ground and nearby objects. To this end, we propose a bucket augmentation method, followed by a segmentation stage consisting of ground trimming and disconnected component trimming. Finally, in order to filter other objects that contain pole-like structures, we introduce the validation process based on the statistical pole descriptor and SVM-based classification.

II. Related Work

Slicing-based Method. Several works use the slicing-based method to deal with point cloud data for finding vertical objects, in order to reduce the influence of structures attached to the vertical trunk. Luo and Wang [2] use slicing to detect pillars from a point cloud. Pu et al. [3] extend the approach of Luo and Wang [2] and propose a percentile-based method to detect the pole-like object. However, their validation method is mainly based on the deviation of the neighboring subparts, which is capable of dealing with pole-like objects with attachments in the bottom or on the top of the trunks, but not enough for pole-like objects with rich attachments in the middle.

Cylindrical Shape-based Method. One of the earliest work aiming at detecting poles from ranged laser data uses Hough voting to detect circles [4]. Similar approaches have been extensively applied in tree trunk detection [5]. However, the circular characteristic for poles is obvious only for indoor environments or close-range scans such as [2].

Segmentation. Due to the presence of ground and facades that connect every objects in a huge cluster, a filtering and segmentation step is usually needed. Yokoyama et al. [6] assumed that the ground in the input data has been removed. Golovinskiy et al. [7] used iterative plane fitting to remove the ground, and a graph-cut-based method to separate the foreground with the background. Tombari et al. [8] used a RANSAC-like iterative algorithm to remove all planar

[1]Jing Huang is with the Department of Computer Science, University of Southern California, Los Angeles, CA 90089, USA huang10@usc.edu

[2]Suya You is with Faculty of the Department of Computer Science, University of Southern California, Los Angeles, CA 90089, USA suya@usc.edu

Fig. 1. Representative poles. From left to right are street lights, flags, utility poles, signs, meters and traffic lights, respectively.

surfaces. We do not, however, rely on global plane fitting in the pre-processing step. Instead, we do the ground trimming after localization, which involves only a small section of the ground.

Point Classification. The point classification is a standard technique to analyze the composition of the point cloud. For example, Lalonde et al. [9] used a Gaussian Mixture Model (GMM) with Expectation Maximization algorithm to learn a model of the three saliency features derived from the eigenvalues of the local covariance matrix, i.e., linear, planar and volumetric. Demantké et al. [10] proposed the dimensionality features based on the same saliency features, which are exhaustively computed at each point and scale. Behley et al. [11] applied spectrally hashed logistic regression to fast classify the points. Hadjiliadis and Stamos [12] proposed a sequential online algorithm for classification between vegetation and non-vegetation and between vertical and horizontal surfaces. Tombari et al.[8] proposed a histogram of scalar products to classify the vertical pole points using SVM. Our method is closest to that of Yokoyama et al. [6], but we make a more delicate classification by distinguishing between wire points and the general linear points.

Pole Classification. Surprisingly, despite the essential trunk, the shapes of the poles are quite diverse. Figure 1 illustrates some representative poles. Most existing works classify the poles according to their usage, shapes and even height. Pu et al. [3] make a detailed classification on the signs according to the planar shapes. However, they do not distinguish between non-planar poles, e.g., lights and utility poles. Golovinskiy et al. [7] make a mixed categorization of usage and height, resulting in seven categories including short post, lamp post, sign, light standard, traffic light, tall post and parking meters, while the utility poles are not reported. Yokohama et al. [6] classify the poles into three categories including street lights, utility poles and signs. In this paper, we follow the categorization of Yokohama et al. [6] since these three categories represent the most common usage of poles, while further classification depending on height could be easily achieved.

III. System Overview

The system pipeline for detection and classification is illustrated in Figure 2. The input is a large-scale point cloud of an urban area, and we output all possible candidates of the pole-like objects and classify them into 3 basic categories, i.e., street lights, utility poles and signs. Specifically, there are three major stages of processing. The first stage is localization, where all possible locations of pole-like objects are extracted. In this stage, we make use of the unique characteristic of pole-like objects, i.e., the local parts of the pole-like objects are also pole-like when they are broken down. The second stage is segmentation, in which the ground and other disconnected components are trimmed at the candidate locations. Finally, we compute the statistical attributes for each candidate based on the extended distribution features and classify the candidates with a support vector machine. The classification step also help filters out other objects that contain local pole-like structures such as trees, pedestrians and part of buildings.

In the following sections, we will discuss each of these stages in detail.

IV. Candidate Localization

We denote the set of pole-like object point clouds as P. Given a scene point cloud Y, any object in it could be seen as a subset of Y, our goal is to find the set of pole-like objects $P_Y = \{P_i | P_i \subset Y \wedge P_i \in P, i = 1, 2, \ldots, n\}$.

Each pole-like object $P_i \in P$ could be divided into two parts: a trunk T_i and the remaining points R_i. The trunk T_i is what makes an object pole-like, while the remaining points R_i can be used to tell what class of pole the object belongs to. We define the location of a pole-like object as the location of its stem.

We focus on the pole-like objects with vertical or near-vertical trunks. If the original input is not well oriented, then a rough ground fitting of the point cloud could be used to generate the upright direction. In fact, most trunks of pole-like objects should not be tilted too much regardless of the terrain.

In the first step, we would like to find all possible locations of pole-like objects. We employ a slicing strategy to obtain a fast raw localization.

A. Point Cloud Slicing and Clustering

One of the important properties of the trunk in the real scene point cloud is that, if it's horizontally sliced, each of its slices would still be a trunk-like segment.

We first slice the input cloud along the z (upright) direction. The height of each layer is $H = 1$ (unit: meter). The slicing result is illustrated in Figure 3 (a).

For each slice, we perform the clustering algorithm based on Euclidean distance to get a list of clusters. The clustering result within each slice is illustrated in Figure 3 (b).

B. Pole Seed Generation

Given a horizontal slice of a pole-like object, there are two obvious characteristics: first, the cross section is relatively small; second, the length is long enough. Moreover, the bounding box of a piece of trunk is very close to the original shape. Therefore, we have the following two criteria regarding the cross section area and the segment length based on the bounding box property of each candidate cluster (Equation 1):

$$\begin{cases} L_x \times L_y < A \\ \qquad L_z \geq \kappa \cdot H \end{cases} \tag{1}$$

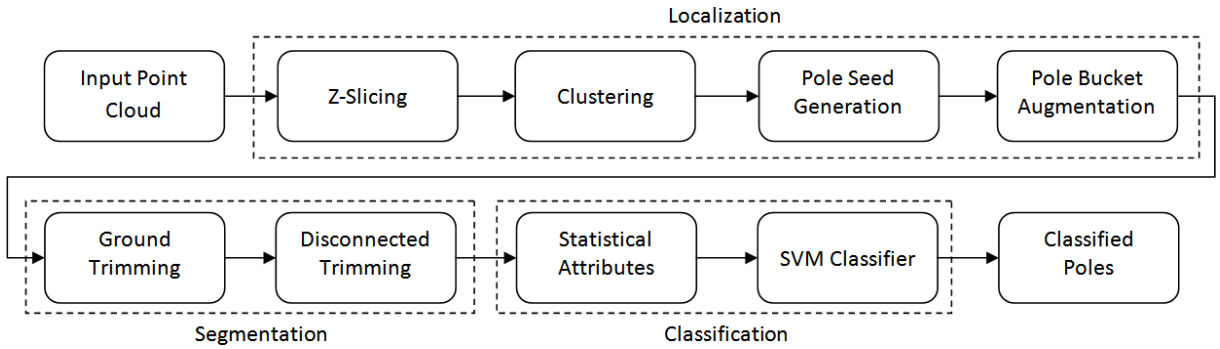


Fig. 2. The pipeline of the proposed pole detection and classification system.

In Equation 1, we assume that the size of the bounding box is $L_x \times L_y \times L_z$, A is the maximum area value of a cross section that would be considered as a trunk, H is the slicing height, and κ is the ratio coefficient of the length. We empirically set $A = 0.7^2 = 0.49$, $H = 1$ and $\kappa = 0.5$. Figure 3 (c) highlights the segments satisfying the trunk criteria.

After all possible trunk segments are generated, we go through them from lower layers to upper layers to check if there is a trunk overlap. Two trunks are said to be overlapping if and only if the bounding boxes of the two trunks projected to the $x - y$ plane have an overlap. Once a trunk overlap is detected, the trunk segment in the upper layer would be added to the trunk list of the lower segment. The lowest trunk segments are considered as the seeds of the poles. This process could extract most candidates even with occlusion, if at least one segment of the trunk is not occluded.

C. Pole Bucket Augmentation

In order to extend the pole candidate from the trunks to their attached structures, we perform the bucket augmentation on the pole seeds. Specifically, the region within a constant horizontal distance $r_b = 3$ from the center of the seed trunk segment is considered as the pole bucket, and all points within this range would be appended to the candidate pole cluster. It's worth noting that, buckets generated by different seeds could overlap and share the similar set of points if the objects are close, but different objects could still be detected independently via the successive steps using the seed information.

V. Candidate Segmentation

While the bucket enhancing limit the range of the pole-like object, the enclosed area does not necessarily belong to the pole. These outliers could possibly be the ground, or points from other objects.

A. Ground Trimming

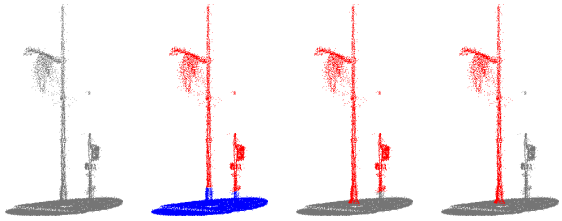
In general, the lowest part of the pole is connected with the cluster of the ground, so it's necessary to trim the ground from the cluster. The idea is that, we can extend the bottom part of the pole as long as the number of points in the inner circle is larger than the outer circle.

Specifically, we only consider the points below the seed trunk, i.e., the ground-connected cluster $G_i = \{p \in C_i | z_p < z_m\}$, where $z_m = \min_{q \in T_i} z_q$ is the lower bound of the z value of the seed trunk. Then, we attempt to extend the bottom of the trunk at a step of $\delta = 0.2$. In the k-th step, we check if inequality (2) holds:

$$\frac{|\{p \in G_{ik} | r_p < r_{inner}\}|}{|\{p \in G_{ik} | r_{inner} < r_p < r_{outer}\}|} < \lambda_G \qquad (2)$$

where G_{ik} is the sliced ground-connected cluster (Equation 3), the inner radius $r_{inner} = \frac{r_b}{4} = 0.75$, the outer radius $r_{outer} = 2r_{inner} = 1.5$, and $\lambda_G = 0.5$ is the trunk-ground ratio. If the inequality holds, the points within the inner radius will be added to the candidate cluster. The process stops when inequality (2) is not satisfied.

$$G_{ik} = \{p \in G_i | z_m - k\delta \leq z_p < z_m - (k-1)\delta\}. \qquad (3)$$



(a) (b) (c) (d)

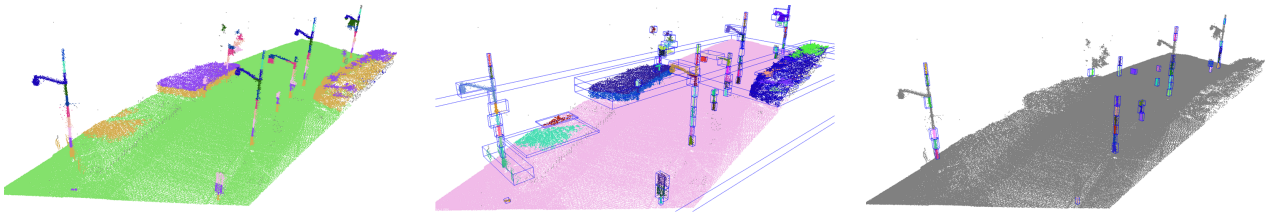
Fig. 4. Ground and disconnected region trimming process. (a) The original bucket-augmented candidate. (b) The red part is the above-the-seed cluster, while the blue part is the under-the-seed, or ground-connected cluster. (c) When ground trimming is done, the bottom part of the pole is successfully extended. (d) Finally, the disconnected components with respect to the seed trunks are removed.

B. Disconnected Region Trimming

To remove the other objects that lie within the range of the bucket, we need to trim the disconnected components. Specifically, we perform the region growing algorithm from each of the seed trunks in the current candidate point cloud, and filter out points that are unreachable. Figure 4 shows the ground and disconnected region trimming process.

VI. Pole Classification

Since the enforced constraints are relatively weak in the localization step so as to keep as many potential poles as



(a) (b) (c)

Fig. 3. The step-by-step results of candidate localization. (a) The sliced result. (b) The clustering result for each slice. (c) The segments satisfying the pole criteria.

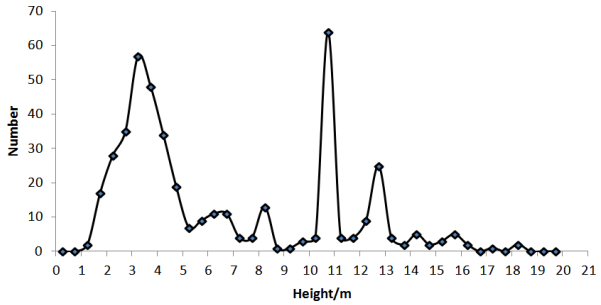


Fig. 5. Distribution of height of poles.

possible. As a side effect, many candidates are actually part of buildings containing pole structures, trees with trunks, and even pedestrians. Therefore, it's necessary to validate them again after segmentation. Meanwhile, most poles belong to lights, utility poles and signs, we try to identify which of these three categories the poles belong to at the same time. To this end, we classify the pole candidates into four categories: lights, utility poles, signs and others. The ones classified as others would be removed from the pole detection result.

Before applying the classifier, we need to extract some attributes from the candidates. The most straightforward attribute about a pole candidate is the height $h = z_{max} - z_{min}$. The height is meaningful because the pole-like objects we would like to detect are man-made objects, thus have fixed height for certain sub-categories. Figure 5 illustrates the distribution of heights of poles from the manually labeled ground truth. There are multiple peaks in different intervals, which suggests a large number of poles of certain types are present in the region.

A. Point Classification

The usage of a pole is highly dependent on its local shapes or attachments, which could be simplified as linear, planar and volumetric components. All pole-like candidates contain a vertical linear trunk. Besides that, lights could contain linear branches or volumetric bulbs, utility poles contain linear wires, and signs contain planar components. These components are further composed of local point-level patches of the same property, so we just need to make a classification on the neighborhood of each point. The traditional Principal Component Analysis (PCA) is applied here.

Suppose that $\lambda_1 \geq \lambda_2 \geq \lambda_3$ are the eigenvalues of the variance-covariance matrix M_p (4):

$$M_p = \frac{1}{|U_p|} \sum_{q \in U_p} (q - \bar{q})(q - \bar{q})^T, \tag{4}$$

where U_p is the set of points lying in the sphere of $r_U = 0.5$ centered at the point p, and $\bar{q}$ is the barycenter of U_p. $\lambda_1 \gg \lambda_2 \simeq \lambda_3$ would indicate there's one principal direction and the distribution is linear; $\lambda_1 \simeq \lambda_2 \gg \lambda_3$ would indicate there are two principal directions and the distribution is planar; $\lambda_1 \simeq \lambda_2 \simeq \lambda_3$ would suggest that there's no obvious principal direction and the distribution is thus volumetric.

There are multiple forms of determining which criterion is satisfied [9], [10], [6]. We apply the form of distribution features as in [6] (5):

$$\begin{cases} S_1 = \lambda_1 - \alpha\lambda_2 \\ S_2 = \lambda_2 - \lambda_3 \\ S_3 = \beta\lambda_3 \end{cases} \tag{5}$$

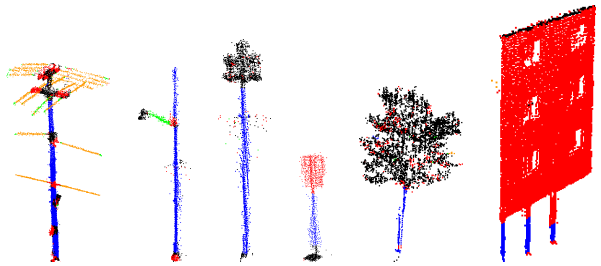
Then, the dimensionality feature $d = argmax_{i \in \{1,2,3\}} S_i$ is introduced to indicate which feature is the most significant. Note that, although Definition (5) does not have a normalization coefficient as in [10], this form is actually more general because only the relative relationship among S_i matters, and the form in [10] corresponds to the case in which $\alpha = 1$ and $\beta = 1$.

In [6], the cluster has been smoothed using the endpoint preserving Laplacian Smoothing in order to recognize linear shapes of different radii. Therefore, very strict parameters are applied for the smoothed data, i.e., $\alpha = 10$ and $\beta = 100$. However, we find that smoothing could cause problems in case of sparse regions (Figure 6), and is time-consuming.

Moreover, without smoothing, a relaxed condition with $\alpha = 4$ and $\beta = 2$ is enough for recognizing most linear shapes and different α could help distinguish the radii of the them. This is particularly useful given the observation that, different from the case in [6], street lights and utility pole cannot be distinguished simply using the proportion of linear points and volumetric points, because both categories can have similar number of linear points. On the other hand, what makes a pole to be a utility pole is that it carries wires.

(a) (b)

Fig. 6. Comparison of point classification result of a façade patch before and after smoothing. (a) shows the classification result without smoothing, in which most points are correctly classified as planar. However, the result with smoothing (b) wrongly classify the planar points as linear points.



(a) (b) (c) (d) (e) (f)

Fig. 7. Point classification results on different objects. The meanings of the colors are: blue - vertical linear, red - planar, black - volumetric, orange - wire, green - other linear.

Therefore, the key is to distinguish the wire points from the other linear points.

While the wire points are classified as linear points in the distribution feature, two distinct properties of wires are enforced: (1) the wires are thinner than other linear parts; (2) the directions of the wires are typically horizontal. To judge the first property, we evaluate a strict condition with $\alpha' = 10$, meaning that S_1' could be larger than S_2 and S_3 only if the linearity is significant enough. The second property requires that principal direction be nearly horizontal, in other words, the eigenvector $\vec{v_1}$ corresponding to λ_1 is roughly perpendicular to the upright direction (Equation 6, $\theta_w = 0.2$).

$$\begin{cases} S_1' = \lambda_1 - \alpha'\lambda_2 > S_2, S_3 \\ |\frac{\vec{v_1}}{||\vec{v_1}||} \cdot (0,0,1)| < \theta_w \end{cases} \tag{6}$$

Similar to [6], the principal direction could be used to distinguish the linear points lying on the vertical trunk by applying the constraint (7) ($\theta_t = 0.8$):

$$|\frac{\vec{v_1}}{||\vec{v_1}||} \cdot (0,0,1)| > \theta_t. \tag{7}$$

Figure 7 shows the classification result of all five categories of points on different types of objects.

B. Pole Component Analysis

From Figure 7 we can qualitatively summarize the relationship between the object classes and the point classification in the following table (I):

TABLE I

QUALITATIVE RELATIONSHIP BETWEEN THE CLASS OF OBJECT AND THE CLASS OF POINT CLASSIFICATION.

Class	Linear			Planar	Volumetric
	Vertical	Wire	Others		
Pole - Light	+	-	-/+	-	-/+
Pole - Utility	+	+	-/+	-	-/+
Pole - Sign	+	-	-/+	+	-
Others - Tree	-/+	-	-	-	++
Others - Façade	-/+	-	-	++	-
Others - Others	-	-/+	-/+	-/+	-/+

In the table, '-' means the object category contains few points of the corresponding class in the column, '+' means the object category contains many points of the corresponding class, '-/+' means the object category could contain few or many points due to different subcategories (e.g. Figure 7(b) and 7(c)), while '++' means the object category contains large number of points. We can see that, without the wire class, it's hard to distinguish between the lights and the utility poles.

From table (I) we can infer some heuristic conditions based on the number of points belonging to different classes to do the classification. However, since the variance within a category could be huge, it's better to apply a learning-based classifier.

In general, we have six attributes for any candidate cluster, including the height h and the number of points in each of the five categories. However, since some joint points on the trunk could be classified as non-linear points, it's better not to count them in the statistics. Similar to the attached part recognition in [6], we apply RANSAC to fit the vertical linear points as trunk. The points lying within distance $\sigma = 0.2$ are considered to be on the fitted line regardless of their class. The process is continued until the number of unfitted vertical linear points is smaller than 50. Another observation is that the non-trunk features on the very bottom of the candidates are typically unrelated (Figure 7). Therefore, we exclude the non-trunk points on the lowest $0.1 \times h$ part of the cluster. Finally, we obtain a refined result for the number of points in each class, i.e., the number of vertical linear points on the fitted trunk n_1, the number of wire points n_2, the number of other linear points n_3, the number of planar points n_4 and the number of volumetric points n_5. Note that n_2, n_3, n_4 and n_5 exclude the points on the fitted trunk and the bottom part of the cluster.

Finally, we normalize them by (8).

$$d_i = \frac{n_i}{N}, i = 1,2,3,4,5. \tag{8}$$

C. Classification by SVM

To classify and validate the poles, we train a 4-class Support Vector Machine (SVM) [13] based on the six attributes. The 4 classes are lights, utility poles, signs and others (non-poles). We use 10-fold validation and grid search for the best C and γ. The training data contain 6 lights, 5 utility poles, 3 signs and 8 instances of non-poles including 4 trees, 3

TABLE II

STATISTICS OF LOCALIZATION.

Number of blocks	45
Number of slices	1287
Number of clusters	114102
Clusters with $H > 0.3$	79225
Clusters with $A < 0.49$	53080
Clusters fitting both criteria	30336
Localized candidates	2448

façade segments and 1 pedestrian. We find that even with such a small group of training dataset, the classification result outperforms the heuristic method. The results are presented in Section VII.

VII. EXPERIMENTS

The scanned data used in our experiment cover a $700m \times 700m$ area of Ottawa from the Wright State 100 dataset [7]. The provided data are merged from one airborne scanner and four car-mounted TITAN scanners facing to the left, right, ground and sky, respectively. The quality of the airborne and TITAN data fusion is 0.05 meters.

We manually labeled 451 poles in the test region containing 45 $100m \times 100m$ blocks of data with over 70 million points. Since many poles are ambiguous in the point cloud, we verify the ground truth with Google Street View. Figure 1 shows a collection of representative poles. The most common category is the light, followed by the sign and utility pole.

Table II shows the statistics for localization. The minimum z value is 40.53 and the maximum z value is 125.76. After slicing in each blocks, there are totally 1287 slices. After clustering within each slice, there are totally 114102 clusters. If only height criterion is applied, there will be 79225 candidate clusters; while if only the area criterion is enforced, there will be 53080 clusters left. 30336 clusters pass both criteria, and after merging the segments, there are 2448 candidate locations.

The 2448 candidates locations are verified through the validation based on the attributes and the SVM classifier, and finally 470 instances are identified as one of the three pole categories (lights, utility poles and signs). Table (III) shows the evaluation of detection of the pole-like objects. Among the 451 pole-like objects, 340 are successfully detected, with a recall rate of 75%. There are 144 false alarms, most of which are trees with small crowns, pedestrians, and sparse building façades with pole-like structures. Table (IV) shows the comparison of pole classification results between our method and that of Yokoyama et al. [6]. Our method turns out to be much better. There are various reasons that the method of [6] fails in this dataset. First, their segmentation step does not take the buildings into account; second, the variation within one category, e.g., the lights are not considered; third, the wires connecting almost all the utility poles make their segmentation unsuccessful; fourth, the smoothing step brings more noises than gains in this dataset; and finally, there are many parameters that need to be tuned in their method, therefore some of the parameters might not fit in this dataset.

TABLE III

EVALUATION OF THE POLE-LIKE OBJECT DETECTION.

Method	Predicted	Correct	Precision	Recall
Yokoyama et al. [6]	483	202	42%	45%
Proposed method	470	340	72%	75%

TABLE IV

EVALUATION OF THE POLE-LIKE OBJECT CLASSIFICATION.

Method	Category	Predicted	Correct	Precision	Recall
[6]	Light	221	73	33%	33%
	Utility	95	2	2%	5%
	Sign	167	27	16%	25%
	Summary	483	102	21%	27%
Proposed	Light	213	119	56%	63%
	Utility	144	72	50%	68%
	Sign	113	54	48%	68%
	Summary	470	245	52%	65%

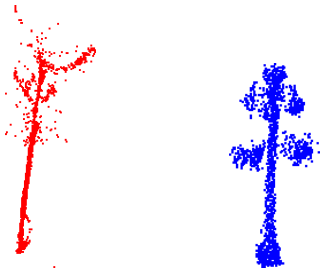
Our method, on the other hand, solve or partially solve the above problems and thus get a superior performance in this challenging dataset.

Figure 8 shows the pole detection and classification result of the whole area and Figure 10 shows some close-ups. The time complexity is generally linear to the number of points in data in segmentation, and most computation needs to be done only on the segmented candidates. The overall running time is around 3 hours, which is fast and demonstrates the feasibility of our method for large-scale urban data. Figure 9 shows the typical failure cases of our method: trees with small crown could be easily confused with the lights with many bulbs. This could possibly be solved by taking more contextual information into account, which is part of our future work.

VIII. CONCLUSION AND FUTURE WORK

In this work, we propose an efficient and easy-to-implement algorithm for pole detection from the 3D scanned point cloud of the urban area. We introduce a series of slicing, combination and filtering strategies, and propose a five-class point classification method to help validate as well

Fig. 8. Pole detection and classification result of the whole area.



(a) (b)

Fig. 9. Typical failure cases of our method. The left is a false alarm due to the sparsity of the tree, while the right is a false negative caused by a light with many bulbs.

as classify the poles. There are some possible improvements in the future work. For example, a plane removal step could be incorporated in pre-processing. Moreover, as indicated in the failure case analysis, the contextual information could be beneficial for figuring out the outliers. Also, we are working on a pole classification method that can further classify the poles into more detailed subcategories.

REFERENCES

[1] P. Doubek, M. Perdoch, J. Matas, and J. Sochman, "Mobile mapping of vertical traffic infrastructure," in *Proceedings of the 13th Computer Vision Winter Workshop*, pp. 115–122, 2008.

[2] D.-a. Luo and Y.-m. Wang, "Rapid extracting pillars by slicing point clouds," in *Proc. XXI ISPRS Congress, IAPRS*, vol. 37, pp. 215–218, Citeseer, 2008.

[3] S. Pu, M. Rutzinger, G. Vosselman, and S. Oude Elberink, "Recognizing basic structures from mobile laser scanning data for road inventory studies," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 66, no. 6, pp. S28–S39, 2011.

[4] P. Press and D. Austin, "Approaches to pole detection using ranged laser data," in *Proceedings of Australasian Conference on Robotics and Automation*, Citeseer, 2004.

[5] T. Aschoff and H. Spiecker, "Algorithms for the automatic detection of trees in laser scanner data," *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 36, no. Part 8, p. W2, 2004.

[6] H. Yokoyama, H. Date, S. Kanai, and H. Takeda, "Detection and classification of pole-like objects from mobile laser scanning data of urban environments," *International Journal of CAD/CAM*, vol. 13, no. 2, 2013.

[7] A. Golovinskiy, V. G. Kim, and T. Funkhouser, "Shape-based recognition of 3d point clouds in urban environments," in *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 2154–2161, IEEE, 2009.

[8] F. Tombari, N. Fioraio, T. Cavallari, S. Salti, A. Petrelli, and L. Di Stefano, "Automatic detection of pole-like structures in 3d urban environments," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pp. 4922–4929, IEEE, 2014.

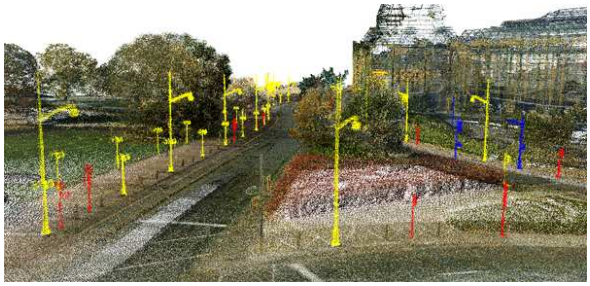
[9] J.-F. Lalonde, N. Vandapel, D. F. Huber, and M. Hebert, "Natural terrain classification using three-dimensional ladar data for ground robot mobility," *Journal of field robotics*, vol. 23, no. 10, pp. 839–861, 2006.

[10] J. Demantké, C. Mallet, N. David, and B. Vallet, "Dimensionality based scale selection in 3d lidar point clouds," *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, Laser Scanning*, 2011.

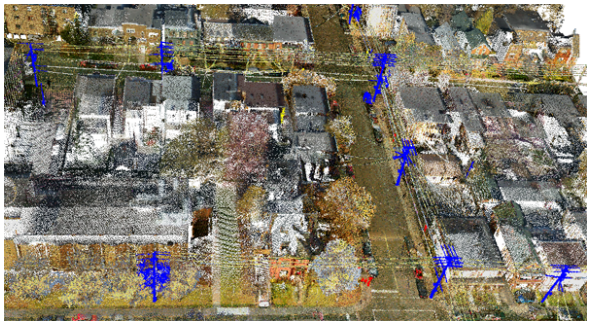
[11] J. Behley, K. Kersting, D. Schulz, V. Steinhage, and A. B. Cremers, "Learning to hash logistic regression for fast 3d scan point classification," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pp. 5960–5965, IEEE, 2010.

[12] O. Hadjiliadis and I. Stamos, "Sequential classification in point clouds of urban scenes," in *Proc. 3DPVT*, 2010.

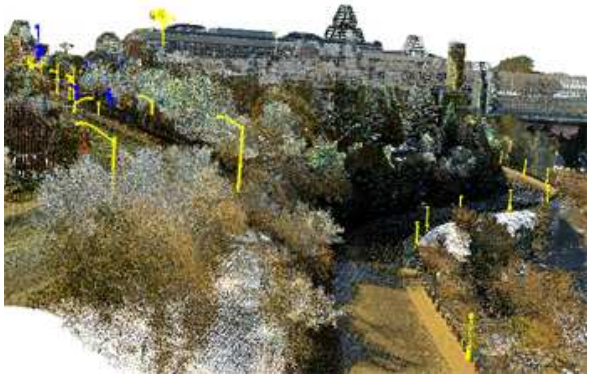
[13] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.



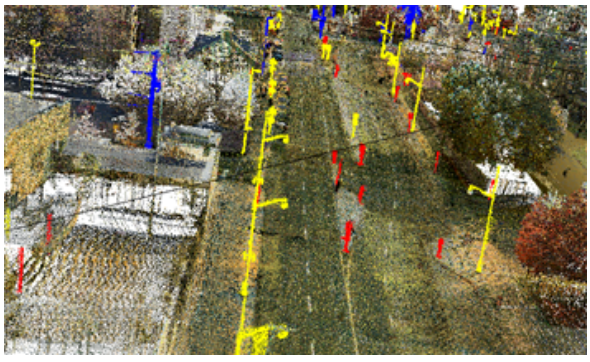
(a)



(b)



(c)



(d)

Fig. 10. Close-ups of pole detection and classification result by our method. We use yellow color to denote the lights, blue color to denote utility poles and red color to denote signs.