

# Power Line Extraction From Mobile LiDAR Point Clouds

Sheng Xu  and Ruisheng Wang 

**Abstract**—This paper proposes a three-step approach to extract power lines from components with mobile laser scanning (MLS) data. First, we use the maximum *a posteriori* estimation to partition point clouds into components. Each component contains points from one object only. The segmentation is optimized by the minimum-cost perfect matching globally and robust to Gaussian noise with the help of the proposed robust estimator. Then, we extract power lines from components based on the linear structure information. Finally, power line components are grouped into individual spans. Experiments show that our method succeeds to achieve the power line extraction from MLS data effectively and outperforms state-of-the-art approaches in terms of the accuracy and robustness.

**Index Terms**—Mobile laser scanning (MLS), power line extraction, robust estimator, segmentation.

## I. INTRODUCTION

**N**OWDAYS, accurately monitoring power lines by LiDAR point clouds has become a high priority for utility companies [1]. The key in this task is the power line extraction from point clouds, which can be achieved by two strategies. One is the classification-based approach [2]–[5], which extracts features first and then chooses a learning model to classify points. The other one is the detection-based approach [6]–[10], which detects candidate points first and then extract power lines based on the line property. The former technique requires a large number of samples for the training process, which is time-consuming and tedious. The latter technique does not contain the learning process but highly depends on the geometric information of a local region, which can be unreliable in the incomplete, unorganized, and uneven mobile laser scanning (MLS) data.

This paper aims to investigate a new approach for power lines extraction from MLS data. Two main contributions are as follows: First, we provide a robust estimator to help partition LiDAR point clouds into components, and second, we succeed to extract power lines from components accurately. The proposed

method achieves optimal segmentation and can be robust to the generated outliers. The geometric information of a single point is based on its neighbor information. However, a local region may contain points from different objects, which increases the challenge in the subsequent linear structure extraction. This is one of the disadvantages in methods based on *k*-nearest neighbors, which is easy to fail in the split of linear structure segments and other objects. Therefore, we extract power lines from components rather than a local region constructed by *k*-nearest neighbors. In this paper, each component belongs to only one object, which is easier for the line extraction.

This paper is organized as follows. Section II reviews the related work on the power line extraction. Section III formulates the segmentation into the maximum *a posteriori* (MAP) estimation and introduces a new robust estimator to against the noise. Section IV presents the optimization of the energy function. Section V extracts power lines from components using the line property information. Section VI evaluates the performance of the power line extraction. Section VII discusses limitations and key points in the proposed extraction. Conclusions are outlined in Section VIII.

## II. RELATED WORK

### A. Power Line Extraction

In the classification-based approaches, Mclaughlin [2] uses the Gaussian mixture model to classify data into transmission lines, vegetation, and other surfaces. Then, the author uses the local affine model to segment transmission lines into individual spans. Liang *et al.* [6] classify the input point clouds into power lines and non-power-lines by examining point's linear structure information first. Then, they calculate the direction of power lines based on the point's distribution information and finally, they use the least square fitting algorithm to reconstruct power lines. Kim and Sohn [3] calculate points' features based on the distribution property first. Then, they use a non-parametric discriminative classifier trained with features to label the point clouds directly. Guo *et al.* [4], [5] calculate features based on the geometry and echo information of the input data first. Then, they use a post-processing segmentation step to separate power lines from others. The missing power lines are addressed by adding the estimation of line model parameters in their post-processing.

In the detection-based approaches, Jwa and Sohn [7] start by extracting candidate power lines based on the line property and then represent power lines through a catenary model. Zhu and Hyppä [8] apply a set of statistical criteria, e.g., the height,

Manuscript received March 1, 2018; revised June 29, 2018 and November 16, 2018; accepted January 8, 2019. Date of publication February 12, 2019; date of current version March 4, 2019. (Corresponding author: Ruisheng Wang.)

S. Xu is with the College of Information Science and Technology, Nanjing Forestry University, Nanjing 210037, China, and also with the Department of Geomatics Engineering, University of Calgary, Calgary, AB T2N 1N4, Canada (e-mail: sheng.xu2@ucalgary.ca).

R. Wang is with the School of Geographical Sciences, Guangzhou University, Guangzhou 510006, China, and also with the Department of Geomatics Engineering, University of Calgary, Calgary, AB T2N 1N4, Canada (e-mail: ruisheng.wang@ucalgary.ca).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/JSTARS.2019.2893967

density, and histogram, for finding candidate power lines first. Then, they transform candidate power lines into a binary image and extract power lines from two-dimensional (2-D) binary images based on the line properties. Cheng *et al.* [10] organize input points into voxels first, and then filter candidate power line voxels based on their geometric features. Finally, they extract power lines by using the polynomial equation fitting. Guan *et al.* [9] apply the elevation difference and slope criteria to detect off-road points first. Then, they extract candidate power lines based on the height, spatial density, and shape information. Finally, they extract power lines by the combination of Hough transform and Euclidean distance clustering.

### B. Point Cloud Segmentation

Since LiDAR point clouds are uneven, unorganized, and massive, points' local information can be not reliable for the calculation of the line property, especially when neighbor points belong to several objects. Therefore, we prefer to extract the linear structure information from components and each component belongs to only one object. However, the accuracy of the segmentation is far from being desired. In the following, typical methods proposed for the point cloud segmentation will be reviewed and analyzed.

The commonly used approach in the segmentation is the clustering. Lavoué *et al.* [11] introduce the classical K-means approach to group points sharing the similar properties. The idea of KMiPC (K-means in point clouds) is to partition points into different sets to minimize the sum of distances of each point in the cluster to the center. The problem is that their results rely on the initial number of sets. Klasing *et al.* [12] present a radially bounded nearest neighbor grouping strategy, which is a kind of  $k$ -nearest neighbors approach. The principle of KNNiPC ( $k$ -nearest neighbors in point clouds) is to select a number of neighbor points based on the distance metric for a given point and assign them the same label. KNNiPC does not require initializing the number of clusters manually as in KMiPC, but the result highly depends on the size of the neighbors. The state-of-the-art methods is to formulate the segmentation as a mathematical optimization problem as shown in Yu *et al.* [13] (3DNCut) and Golovinskiy *et al.* [14] (MinCut).

3DNCut succeeds to extend the 2-D normalized-cut [15] for 3-D point cloud segmentation by adding the elevation information. It partitions input points into two disjoint groups by minimizing the dissimilarity within each group and maximizing the dissimilarity between different groups. Each point will be regarded as a node in the formulation of their graph and spatially close nodes are connected with each other. Connections in the graph are weighed by the Euclidean distances between points. 3DNCut succeeds to obtain optimal results for the problem of two-label segmentation. But similar to KMiPC, a predefined parameter for the number of objects has to be set manually in the multi-object segmentation.

MinCut partitions input points into two disjoint groups, i.e., the background and foreground, by minimizing the sum of the data term and the smoothness term. The data term measures how appropriate a label is for a point given the observed data, and the

smoothness term is to constrain labels of the neighboring points. For the graph model, points are formed as nodes, and neighboring points are connected with each other. Each point connects with both sink vertex (background) and source vertex (foreground). MinCut uses the Euclidean distances between neighboring points as weights for connections. The edges between nodes and the sink vertex are weighed by a user-defined value. The edges between nodes and the source vertex are weighed by the Euclidean distances between nodes and the center point. The solution cut, which separates the graph into background and foreground, is obtained by the graph-cut method [16]. MinCut obtains competitive segmentation results in terms of the optimum and accuracy. However, weights of connections between nodes and vertices rely on the radius and the center point of each object.

The above-mentioned shortcomings make them difficult to segment multi-object from point clouds. Moreover, their robustness to noise is low. This is because their optimization process ignores the contamination generated during the LiDAR point collection. Therefore, our target is to propose a new method to address the existing disadvantages in the LiDAR point cloud segmentation first. Then, extract power liners from components which contain accurate geometric information, i.e., the linear structure.

## III. ENERGY FUNCTION FORMULATION

### A. Segmentation Model Derivation

Assume that  $C = \{c_1, c_2, \dots, c_n\}$  is the set of input points and  $L = \{1, 2, \dots, m\}$  is the set of component labels. The component label of a point  $c_i$  is denoted by  $l_i$  and  $l_i \in L$ . From Bayes' rule, the joint posterior probability over  $L$  given  $C$  is

$$P(L | C) = \frac{P(C | L) P(L)}{P(C)}. \quad (1)$$

In this paper, we define

$$P(C | L) \propto \prod_{c_i \in C} \exp(-\varphi(c_i, l_i)) \quad (2)$$

where  $\varphi(c_i, l_i)$  is the data term to calculate the penalty of assigning a point  $c_i$  with the label  $l_i$ . Under the assumption that the label of a point depends only on labels of its neighborhood, we have

$$P(L) \propto \prod_{c_i \in C} \prod_{c_j \in N(i)} \exp(-\psi(c_i, c_j, l_i, l_j)) \quad (3)$$

where  $N(i)$  is the set of neighbor points of  $c_i$ , and  $\psi(c_i, c_j, l_i, l_j)$  is the smoothness term to constrain the consistency of neighbor points' labels. Combine (1)–(3), we have

$$P(L | C) \propto \exp\left(-\sum_{c_i \in C} \varphi(c_i, l_i)\right) \exp\left(-\sum_{c_i \in C} \sum_{c_j \in N(i)} \psi(c_i, c_j, l_i, l_j)\right).$$

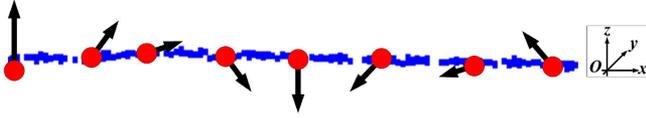


Fig. 1. Inconsistent normal vectors.

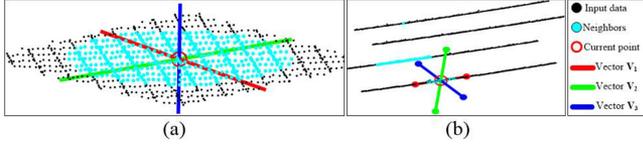


Fig. 2. Illustration of the decomposed vectors at a point. (a) Vectors at a plane region. (b) Vectors at a line region.

This paper models the segmentation as an MAP estimation and the optimal segmentation label configuration  $\hat{L}$  becomes

$$\begin{aligned} \hat{L} &= \arg \max_L P(L | C) \\ &= \arg \min_L \left( \sum_{c_i \in C} \varphi(c_i, l_i) + \sum_{c_i \in C} \sum_{c_j \in N(i)} \psi(c_i, c_j, l_i, l_j) \right). \end{aligned} \quad (4)$$

In the initialization, each point has a unique component label. The value of (4) is large, because points in the homogeneous region are assigned with different labels. Our goal is to update labels of points to optimize (4). The following is to show the calculation detail of the data term and smoothness term.

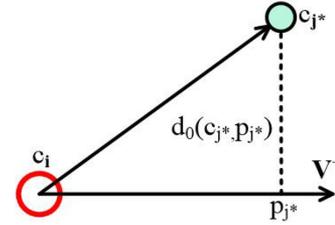
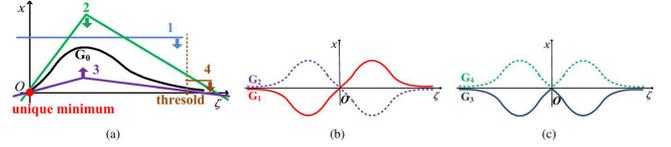
As we know, homogeneous points in a local plane usually share the similar normal vector. However, normal vectors at points from lines are inconsistent, as shown in Fig. 1. Therefore, our data term consists of both the normal vector  $\mathbf{V}_i^{\perp}$  and the principal direction vector  $\mathbf{V}_i^{-}$  as

$$\varphi(c_i, l_i) = (1 - \lambda_i) \cdot \rho(\mathbf{V}_i^{\perp}, \mathbf{V}_{j^*}^{\perp}) + \lambda_i \cdot \rho(\mathbf{V}_i^{-}, \mathbf{V}_{j^*}^{-}) \quad (5)$$

where  $\mathbf{V}_{j^*}^{\perp}$  and  $\mathbf{V}_{j^*}^{-}$  are the normal vector and principal direction vector at the point  $c_{j^*} \in (\Omega(i) \cap N(i))$ , respectively.  $\Omega(i)$  is the set of points sharing the label  $l_i$ . In this paper, we approximate the normal vector  $\mathbf{V}_i^{\perp}$  and the principal direction vector  $\mathbf{V}_i^{-}$  to a given point  $c_i$  by performing Principal component analysis (PCA) [17] on the covariance matrix of points in  $\Omega(i) \cap N(i)$ . Fig. 2(a) and (b) illustrates the decomposed vectors by PCA at a point in a plane and in a line, respectively.  $\mathbf{V}_i^{-}$  refers to the largest component  $\mathbf{V}_1$  and  $\mathbf{V}_i^{\perp}$  corresponds to the smallest component  $\mathbf{V}_3$ , respectively.  $\lambda_i$  aims to balance the normal vector and principal direction vector, which is calculated as

$$\lambda_i = \frac{1}{|\Omega(i) \cap N(i)|} \sum_{c_{j^*}} d_0(c_{j^*}, p_{j^*}) \quad (6)$$

where the point  $p_{j^*}$  is the projection of  $c_{j^*}$  on the vector  $\mathbf{V}_{j^*}^{-}$ , as shown in Fig. 3.  $d_0(c_{j^*}, p_{j^*})$  decides whether a point belongs to a line or not depending on a threshold  $d_l$ . If the Euclidean distances between the point  $c_{j^*}$  and  $p_{j^*}$  is smaller than  $d_l$ ,  $d_0(c_{j^*}, p_{j^*})$  will be 1, otherwise  $d_0(c_{j^*}, p_{j^*})$  will be 0.

Fig. 3. Calculation of the coefficient  $\lambda_i$  in the data term.Fig. 4. Plot of a robust influence function. (a) Requirements of the robust estimator. (b) Center-symmetric type  $G_1$  and  $G_2$ . (c) Axis-symmetric type  $G_3$  and  $G_4$ .

"|" measures the cardinality of a set. If the point  $c_i$  is from a line region,  $\lambda_i$  will be close to 1 and the principal direction determines  $\varphi(c_i, l_i)$  in (5). If the point  $c_i$  is from a plane region,  $\lambda_i$  will be close to 0 and the normal vector decides the data term. The target of (5) is to keep the normal vector and the direction consistent in a local region.

The smoothness term  $\psi(c_i, c_j, l_i, l_j)$  is defined as

$$\psi(c_i, c_j, l_i, l_j) = \rho(c_i, c_j) \cdot \delta_{l_i \neq l_j} \quad (7)$$

where  $\delta_{l_i \neq l_j}$  is a binary number. If  $l_i \neq l_j$ ,  $\delta_{l_i \neq l_j}$  is 1, otherwise  $\delta_{l_i \neq l_j}$  is zero. The target of (7) is to segment spatially close points into the same group using the Euclidean distances between points.

The above-mentioned  $\rho$  is an estimator to evaluate the distance between vectors [e.g., in (5)] or points [e.g., in (7)], which will be discussed in Section III-B.

## B. Robust Estimator Formulation

To deal with the contaminated LiDAR data, we introduce the robust estimation by analyzing the influence function of estimators to measure the difference between points' properties. The influence function shows the infinitesimal behavior of the asymptotic value and measures the asymptotic bias caused by the contamination [18]. To be a robust estimator, as shown in Fig. 4(a), the influence  $\chi$  based on the error  $\zeta$  is supposed to be as follows:

- 1) lower than the curve 1 to be bounded;
- 2) lower than the curve 2 to increase or decrease gently to the limit;
- 3) higher than the curve 3 to increase or decrease effectively;
- 4) lower than the curve 4 to be vanished beyond a threshold;
- 5) 0 at the origin to achieve the unique minimum.

As shown in Fig. 4(a), the curve  $G_0$  meets all the above requirements, which can be regarded as a part of a robust influence function. The entire influence function  $G(\zeta)$  can be designed in center- or axis- symmetry of  $G_0$ , as shown in Fig. 4(b) and (c).

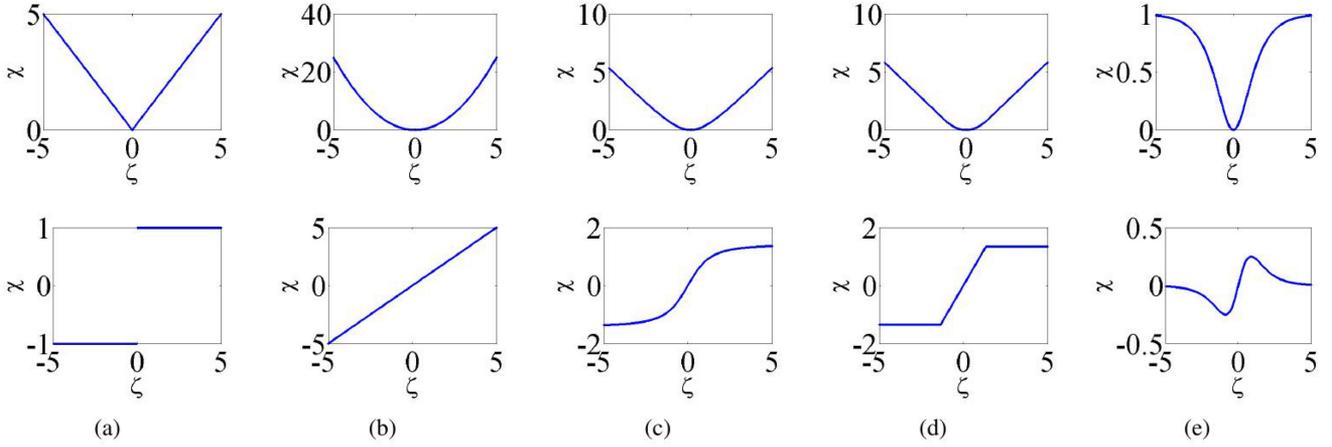


Fig. 5. Estimators and influence functions. Estimators are in the first row and their corresponding influence functions are in the second row. (a) Function of  $L_1$  estimator:  $\rho_{L_1} = |\zeta|$  and its influence function:  $\rho_{l_1} = \text{sign}(\zeta)$ . (b) Function of  $L_2$  estimator:  $\rho_{L_2} = \zeta^2/2$  and its influence function:  $\rho_{l_2} = \zeta$ . (c) Function of  $L_1 - L_2$  estimator:  $\rho_{L_1-L_2} = 2 \left( \sqrt{1 + \zeta^2/2} - 1 \right)$  and its influence function:  $\rho_{-} = \frac{\zeta}{\sqrt{1 + \zeta^2/2}}$ . (d) Function of Huber's estimator:  $\rho_H = \zeta^2/2$ , if  $|\zeta| < r$ ;  $\rho_H = r(|\zeta| - r/2)$ , if  $|\zeta| \geq r$ ,  $r = 1.345$  and its influence function:  $\rho_h = \zeta$ , if  $|\zeta| < r$ ;  $\rho_h = r \cdot \text{sign}(\zeta)$ , if  $|\zeta| \geq r$ ,  $r = 1.345$ . (e) Function of the proposed estimator:  $\rho_O = 1 - \frac{2}{e^\zeta + e^{-\zeta}}$  and its influence function:  $\rho_o = \frac{e^\zeta - e^{-\zeta}}{(e^\zeta + e^{-\zeta})^2}$ .

Fig. 5 shows five estimators and their influence functions in the first and second row, respectively. Commonly used non-robust estimators are the  $L_1$  estimator:  $\rho_{L_1}$  and the  $L_2$  estimator:  $\rho_{L_2}$ , as shown in Fig. 5(a) and (b), respectively. Their problems are that  $\rho_{L_1}$ 's influence function  $\rho_{l_1}$  is not strictly continuous at  $\zeta = 0$ , which may cause an indeterminate result and  $\rho_{L_2}$ 's influence function  $\rho_{l_2}$  is not bounded. Two better estimators are the  $L_1 - L_2$  estimator:  $\rho_{L_1-L_2}$  and the Huber's estimator [19]:  $\rho_H$ , as shown in Fig. 5(c) and (d), respectively. The plot of the estimator  $\rho_{L_1-L_2}$  and  $\rho_H$  are with the same shape and amplitude. However, they have different influence functions, i.e.,  $\rho_{-}$  is smoother than  $\rho_h$ . This means that it is difficult to analyze the robustness of an estimator without the help of its influence function. From their influence functions  $\rho_{-}$  and  $\rho_h$ , we know that they can reduce the influence of large errors and keep stable at  $\zeta = 0$ , however, they are not vanished after arriving at the maximum. Fig. 5(e) shows the plot of the proposed robust estimator  $\rho_O$  defined as  $1 - \frac{2}{e^\zeta + e^{-\zeta}}$  and the influence function  $\rho_o$  calculated by  $\frac{e^\zeta - e^{-\zeta}}{(e^\zeta + e^{-\zeta})^2}$ . Our influence function  $\rho_o$  is formed based on the proposed influence function  $G_1$  and overcomes all of the above-mentioned drawbacks.

It is worth noting that the amplitude of the proposed estimator can be adjusted by multiplying a positive number. In this paper, we scale the difference of points and normal vectors from 0 to 1, as shown in the first row of Fig. 5(e). Therefore, the estimation of properties' difference by the above proposed robust estimator can be defined as

$$\rho(*, *) = 1 - \frac{2}{e^\zeta + e^{-\zeta}}. \quad (8)$$

If  $\rho(*, *)$  measures differences between two vectors  $\mathbf{V}_i$  and  $\mathbf{V}_j$  as shown in (5), we have  $\zeta = \sin(\angle(\mathbf{V}_i, \mathbf{V}_j))$ , i.e., the sine of the angle between the vector  $\mathbf{V}_i$  and  $\mathbf{V}_j$ . If  $\rho(*, *)$  measures differences between two points  $c_i$  and  $c_j$  as shown in (7), we

have  $\zeta = \frac{1}{\|c_i - c_j\|}$ , i.e., inversely proportional to the Euclidean distances between the point  $c_i$  and  $c_j$ .

#### IV. MINIMUM-COST PERFECT MATCHING (MCPM) MODEL

Equation (4) can be solved by many existing optimization methods, e.g., the graph-cut and normalized-cut approach. However, we expect to achieve the multi-object segmentation without any human-computer interaction as in graph-cut and without initializing the number of objects as in normalized-cut. Since the graph-based optimization works well in flexibly formulating the problem constraints and globally achieving the optimal solution in the minimization [20], this section aims to solve the energy function optimization using the bipartite graph.

Our idea is that, a high-energy function can be minimized by updating neighbor points' labels. Therefore, the key of the optimization is to update the label of each point (label configuration) optimally. For an input point set  $C = \{c_1, c_2, c_3, \dots, c_n\}$ , we define a matrix  $\mathbf{M}_{n \times n}$  to form the objective function for solving the optimal update solution. The element  $\mathbf{M}(i, j)$  is a binary value. If  $\mathbf{M}(i, j)$  is 1, points in  $\Omega(i) \cup \Omega(j)$  will be updated as sharing the same label, where  $\Omega(i)$  is the set of points sharing the label  $i$ . In the initial label configuration, each unique label contains only one point, i.e.,  $\Omega(1) = \{c_1\}$ ,  $\Omega(2) = \{c_2\}$ ,  $\Omega(3) = \{c_3\}$ ,  $\dots$ ,  $\Omega(n) = \{c_n\}$ . If the minimum Euclidean distances between  $\Omega(i)$  and  $\Omega(j)$  is small ( $\leq d_p$ ),  $\mathbf{M}(i, j)$  will be initialized as 1, otherwise  $\mathbf{M}(i, j)$  will be 0. Neighbor points with different labels will be updated into the same label when they are homogeneous. Our goal is to iteratively update the labels of points to decrease the energy in (4) until the label configuration converges, which means that homogeneous points share the same component label. In this paper, each point has only one label, i.e.,  $\Omega(i) \cap \Omega(j) = \emptyset$ . Points sharing the same label will be regarded as in the same component. The objective function and constraints for optimizing the update solution are defined

as

$$\begin{aligned} \text{Minimize}_{\mathbf{M}} \quad & E(\mathbf{M}) = \sum_{i=1}^n \sum_{j=1}^n \mathbf{M}(i, j) \cdot \gamma(i, j) \\ \text{Subject to} \quad & \sum_{i=1}^n \mathbf{M}(i, t) = 1, t \in \{1, 2, 3, \dots, n\} \\ & \sum_{j=1}^n \mathbf{M}(t, j) = 1, t \in \{1, 2, 3, \dots, n\} \end{aligned} \quad (9)$$

where the weight function  $\gamma(i, j)$  is calculated as

$$\gamma(i, j) = \begin{cases} \alpha, & i = j \\ \varphi(c_i, l_i) + \psi(c_i, c_j, l_i, l_j), & i \neq j. \end{cases} \quad (10)$$

If  $\mathbf{M}(i, j)$  is 1, there may be a combination between the  $i$ th and  $j$ th component. The penalty for the combination is calculated by  $\gamma$ . If the  $i$ th component is not merged with any other component, there will be a user-defined penalty  $\alpha$ . Two constraints in (9) shows that there is only one element that can be assigned with 1 in each row or column of  $\mathbf{M}$ , i.e., each component can only be merged with one component. In the work of Xu *et al.* [21], they only consider the normal vector information in the cluster combination, therefore, most power lines are removed as isolated clusters. In this paper, the principal direction is included into the segmentation optimization. The following aims to show how to minimize (9) for the segmentation based on the McPM.

The defined  $\mathbf{M}$  is a symmetric matrix that can be represented by a bipartite graph  $\mathbb{G} = \{\mathbb{V}_x, \mathbb{V}_y, \mathbb{E}\}$ . The node sets  $\mathbb{V}_x = \{\Omega_1^x, \Omega_2^x, \dots, \Omega_i^x\}$  and  $\mathbb{V}_y = \{\Omega_1^y, \Omega_2^y, \dots, \Omega_j^y\}$  represent the point set of each label and  $\Omega_i^x = \Omega_i^y = \Omega(i)$ . The edge set  $\mathbb{E} = \{e_{1,1}, e_{1,2}, e_{1,3}, \dots, e_{i,j}\}$  indicates the connection between  $\mathbb{V}_y$  and  $\mathbb{V}_x$  for restricting the label update process. As we mentioned before, each row or column has only one element that are equal to 1 as shown in (9). This constraint can be achieved by the perfect matching in the bipartite graph, which is defined as each node has a matching, i.e., a set of edges without common vertices in a graph. Therefore, the solution of (9), which determines an update solution, can be solved by the corresponding perfect matching in  $\mathbb{G}$ . Connection between  $\Omega_i^x \in \mathbb{V}_x$  and  $\Omega_j^y \in \mathbb{V}_y$  in the perfect matching means that points in  $(\Omega_i^x \cup \Omega_j^y)$  will be updated as sharing the same label. We weigh the edge  $e_{i,j}$  by the above-mentioned function  $\gamma(i, j)$  and the optimal update solution can be achieved by solving the McPM of  $\mathbb{G}$ .

## V. POWER LINES EXTRACTION FROM COMPONENTS

After we partition input point clouds into optimal components, each component will only contain unique geometric information. However, a power line is usually split into different line components, as shown in Fig. 6(a). Xia and Wang [22] propose a linear structure clustering method based on the  $k$ -nearest neighbor graph for MLS point clouds. However, their method relies on the gradients in point clouds, which are not applicable to power lines. The following aims to group power lines into individual spans by analyzing the line property of components.

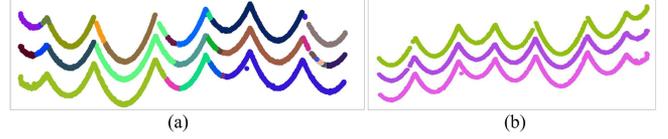


Fig. 6. Line components and the merging results. (a) Extracted line components. (b) Three complete individual spans.

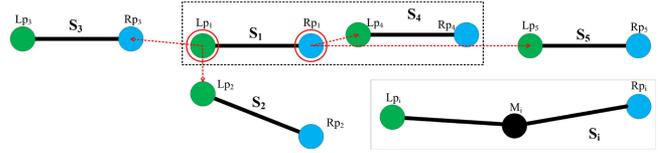


Fig. 7. Illustration of the line merging process.

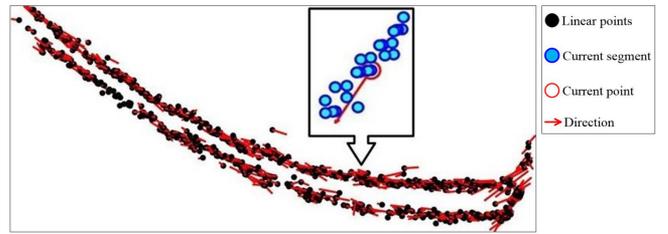


Fig. 8. Principal direction at each point.

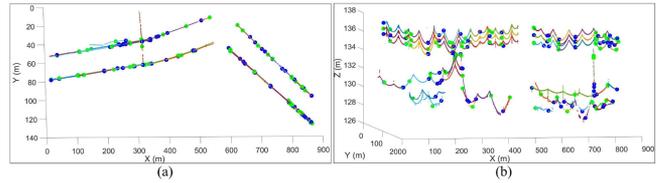


Fig. 9. Different views of the left and right points in line components. (a) Top-down view. (b) Side view.

As shown in Fig. 7, for a component  $S_i$ , we denote its mean coordinate point as  $M_i$  and we find the furthest point on each side of  $M_i$  as  $Lp_i$  and  $Rp_i$ , respectively. The line merging is based on the minimum distances between components. For each component  $S_i$ , we calculate the principal direction of each point, as shown in Fig. 8. If the component  $S_i$  and  $S_j$  are in the same direction ( $\leq d_d$ ), the minimum Euclidean distances between the component  $S_i$  and  $S_j$  is defined as

$$D(S_i, S_j) = \min(\|Rp_j - Lp_i\|, \|Lp_j - Lp_i\|, \|Rp_j - Rp_i\|, \|Lp_j - Rp_i\|). \quad (11)$$

In the example shown in Fig. 7, the component  $S_1$  will be merged with its closest component  $S_4$ .  $S_3$  and  $S_5$  are too far from  $S_1$  in terms of the Euclidean distance.  $S_2$  and  $S_1$  are not in the same direction. After the line merging, we will update the left and right points for each component based on the new middle point. Fig. 9(a) and (b) shows the left and right points in the line merging from different views. In the merging of line components in Fig. 6(a), we achieve three complete individual spans, as shown in Fig. 6(b).

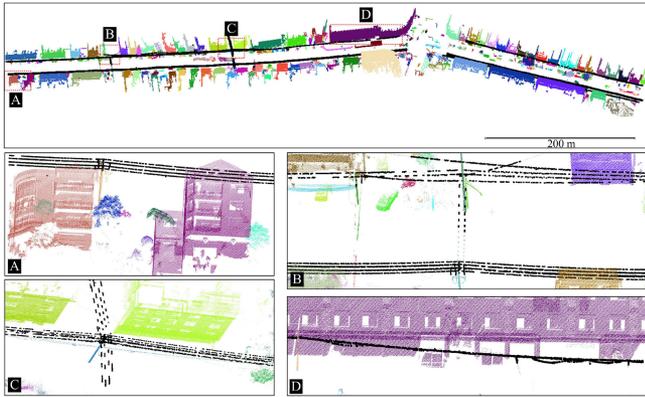


Fig. 10. Power line extraction results.

## VI. EXPERIMENT AND EVALUATION

### A. Power Line Extraction Results

This section aims to show the performance of our power line extraction, including splitting input point clouds into components and grouping power lines into individual spans. The experimental scene contains 6 124 105 points and the area is over 200 m by 800 m. Fig. 10 shows the segmentation result based on the bipartite graph optimization. Individual components are illustrated by different colors. Although errors appear in the overlapping region between different vegetations, power lines (black) are detected effectively based on the line's property information. We show four close views of different power line cases. Fig. 10(a) shows the extraction results of the straight power lines. Fig. 10(b) shows the extraction results of the incomplete power lines. Fig. 10(c) shows the extraction results of the crossing power lines. Fig. 10(d) shows the extraction results of the twined power lines. In all test cases, we succeed to extract power lines from the achieved components.

Results of the individual span grouping are shown in Fig. 11. Fig. 11(a) is the top-down view of all extracted power line spans. Fig. 11(b) shows results of the straight power lines. Fig. 11(c) shows results of the crossing power lines. Fig. 11(d) shows results of the power lines in various densities. Fig. 11(e) shows results of the power lines in the complex situation, including the twined power lines and spatially close power lines. As shown in Fig. 11, the proposed line component grouping succeeds to achieve all individual spans in the experimental scene.

### B. Extraction Performance Evaluation

The evaluation of our power line extraction requires the calculation of true positive (TP), false negative (FN), and false positive (FP). TP means that power lines are detected correctly from the input. FN means that power lines are wrongly detected as non-power-lines. FP means that non-power-lines are wrongly detected as power lines. The ground truth of power lines for the reference is obtained manually from the input. To evaluate extraction results, we calculate the correctness, completeness, and

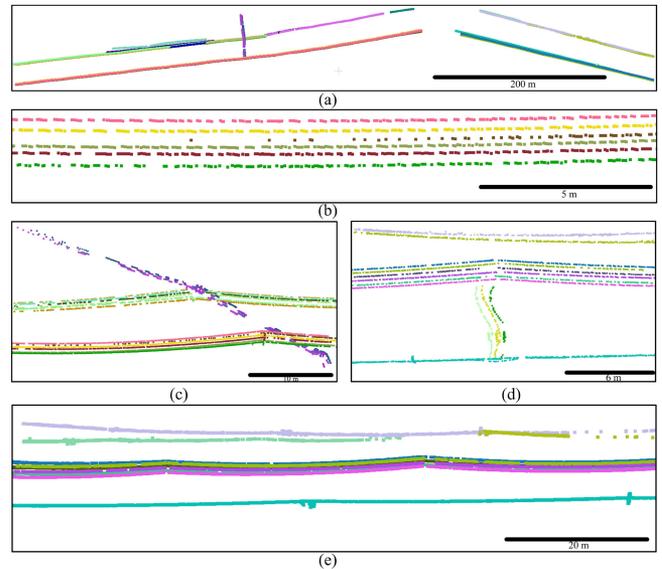


Fig. 11. Individual spans. (a) Top-down view of results. (b) Straight power lines. (c) Crossing power lines. (d) Incomplete power lines. (e) Complex power lines.

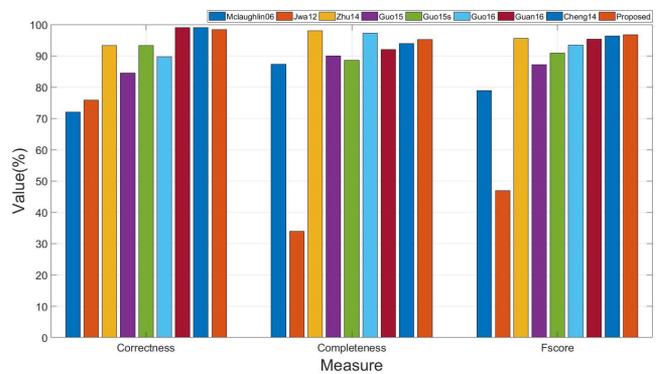


Fig. 12. Comparison with other power line extraction methods.

Fscore as

$$\begin{aligned} \text{Correctness} &= \frac{TP}{TP + FP} \\ \text{Completeness} &= \frac{TP}{TP + FN} \\ \text{Fscore} &= \frac{2 \times TP}{2 \times TP + FP + FN} \end{aligned} \quad (12)$$

respectively. The correctness measures the ratio of correctly extracted power lines in results, and the completeness measures the percentage of correctly extracted power lines in ground truth. The Fscore is the harmonic mean of correctness and completeness. In order to demonstrate the superiority of the proposed method, we compare our results with other methods, including Mclaughlin06 [2], Jwa12 [7], Zhu14 [8], Guo15 [4], Guo16 [5], Guan16 [9], and Cheng14 [10] in Fig. 12.

Our correctness is a little lower than Guan16 [9] and Cheng14 [10], this is because our data contain lots of overlapping

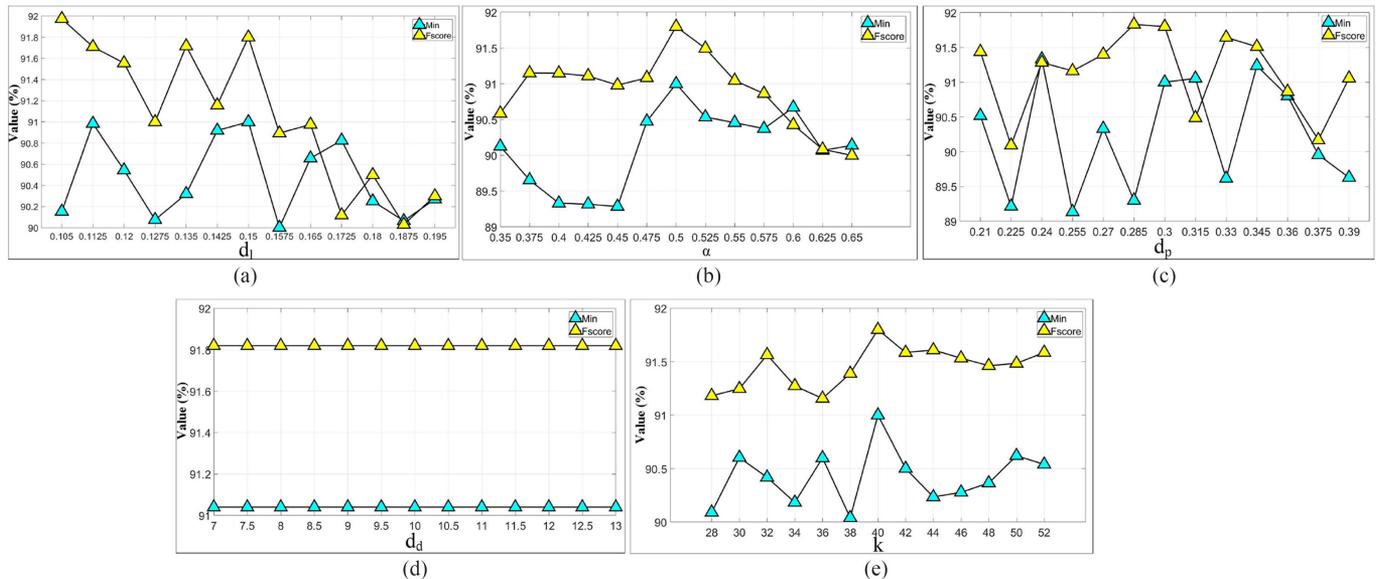


Fig. 13. Evaluation of different parameters.

TABLE I  
EVALUATION RESULTS WITH DIFFERENT PARAMETERS

Parameters	Suggested	Unit	Mean accuracy(%)	
			Min(Corr.,Comp.)	Fscore
$d_l$	0.15	meter	90.46	91.05
$\alpha$	0.5	N/A	90.11	90.90
$d_p$	0.3	meter	90.24	91.13
$d_d$	10	degree	91.04	91.82
$k$	40	point	90.43	91.45

regions between power lines and vegetations. Our completeness is lower than Zhu14 [8] and Guo16 [5], this is because our experimental scenes are much more complex. Recently, Chen *et al.* [23] propose an automatic method to extract transmission lines from unmanned aerial vehicle LiDAR data. Both correctness and completeness of their method are better than ours. However, their method depends on the catenary curve fitting and fails to segment the top of the tree encroachment in the case of the low point density. As shown in Fig. 10(b), our work performs effectively and accurately on the extraction of incomplete power lines. The evaluation of Fscore shows that the proposed method achieves a good balance between the correctness and completeness.

### C. Parameters Analysis

There are five parameters in the proposed power line extraction method. The suggested parameter values are given in Table I.  $d_l$  is used in the data term for  $\lambda$ , which measures the Euclidean distances between a point and a line. If a point is close to a line, this point will be regarded as belonging to this line.  $\alpha$  is a user-defined penalty for the segmentation. Only if the penalty of merging two components is smaller than  $\alpha$ , these two components can be merged. A large  $\alpha$  will increase the ratio of the under-segmentation and a small  $\alpha$  will increase the ratio of the over-segmentation.  $d_p$  and  $k$  are used for the selection of the neighbor components and points, respectively. In this paper,

$d_p$  is not sensitive to different densities, because the Euclidean distances between two individual spans are much larger than two points.  $k$  is used in the calculation of normal vector and principal direction, which has to be tuned for different scenes. A large  $k$  will cause errors in the linear structure calculation and increase the number of false negative power lines.  $d_d$  is used for grouping power lines into individual spans. Only if the angle between two lines is smaller than  $d_d$ , these two lines can be regarded as lying in the same line. A large  $d_p$  and  $d_d$  will make it difficult to group power lines into individual spans.

In order to show how a variation in the above-mentioned parameters will affect the results, we conduct the sensitivity analysis. For the purpose of the analysis, we alter all parameters from  $-30\%$  to  $30\%$  with respect to the suggested values. The analysis is conducted by floating one parameter and fixing the rest of parameters. The accuracy of the above-mentioned scenes with different parameters is shown in Fig. 13. The accuracy is stable in the case of floating  $d_d$  and slightly degraded in the case of floating  $\alpha$  and  $k$ . Although the accuracy in the case of floating  $d_l$  and  $d_p$  are not stable, we can still achieve a promising average accuracy. In each case, the minimum of the correctness and completeness is no less than 90.11% and the Fscore is larger than 90.90%, as given in the last two columns of Table I.

## VII. DISCUSSION

### A. Performance on the Large-Scale Data

The complexity of the segmentation depends on the solving of McPM in the bipartite graph using the Kuhn–Munkres algorithm [24], which is  $\mathcal{O}(n^2)$ . This can be a drawback in the extraction from large-scale point cloud sets.

This experimental scene contains 272 618 869 points and the area is over 763 m by 4359 m. Since most power lines are along the roadside, we let the region of interest (ROI) be the area 15 m to the middle of the road in this experiment. The road information can be obtained from the trajectory. Since it

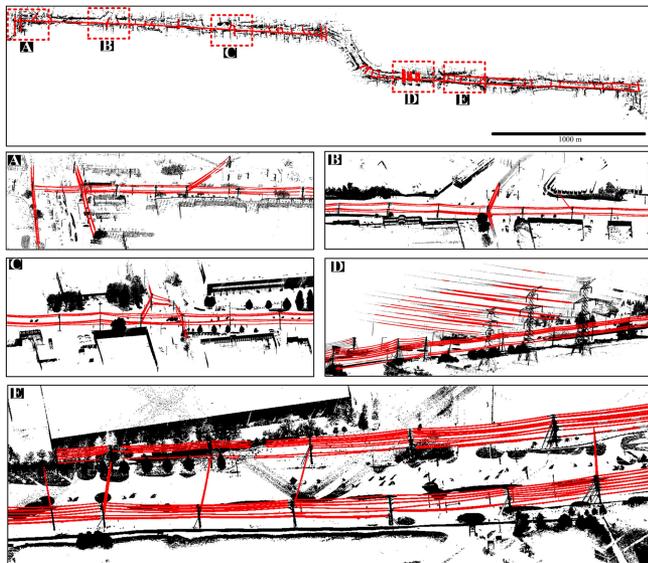


Fig. 14. Power line extraction from a large-scale point cloud set.

is an urban scene, the elevation of power lines is assumed to be high. Therefore, components below 2 m will be regarded as non-power-lines. Besides, if a point set  $\Omega_i^x$  contains more than 10 000 points and less than 50% of points are lying in the same line,  $\Omega_i^x$  will be labeled as a non-power-line component and removed in the current iteration. In this way, all non-power-lines are grouped together (black) and the rest of input points are regarded as power lines (red), as shown in Fig. 14.

Fig. 14(a)–(e) are close-views of the extraction results. All power lines in Fig. 14(a), (c), and (e) are extracted. A short part of power lines is missing in Fig. 14(b), this is because that it is not in the ROI. The top region of power lines in Fig. 14(d) is not complete, this is because that the sag of power lines is heavy in the high voltage line. Power lines are segmented into very small pieces and regarded as non-power-lines. Although parts of power lines are missing in B and D, we succeed to extract most power lines.

In this experiment, we do not split power lines into individual spans for the purpose of efficient extraction. Experiments were done on a Windows computer, Intel Core i7-6900k, 3.20 GHz processor with 64 GB of RAM. The run time in the extraction is 98.56 s excluding the time for data loading.

### B. Discussion of the Proposed Segmentation

*How does the segmentation work?* In the beginning, each unique label contains only one point which means that each point is a component. The number of labels in  $\mathbb{V}_x$  is  $n$ , which will be decreased during the update process. In order to converge to a stable label configuration, we proceed the update process iteratively. The number of the iteration ranges from 0 to  $n - 1$ , and in each iteration, the proposed optimization updates labels at the minimum cost. The segmentation result is determined by perfect matchings of the bipartite graph. The capacity of the perfect matching is equal to the energy in (9). Therefore, each segmentation result can be determined by a perfect matching

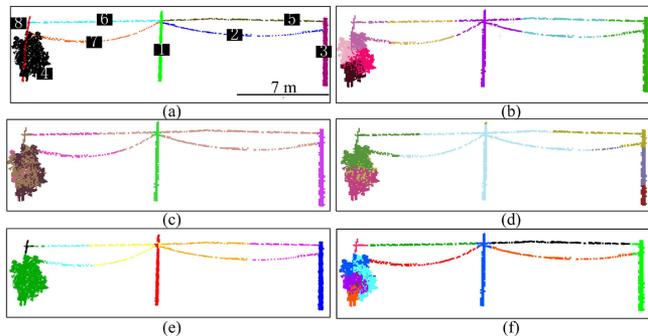


Fig. 15. Performance of different segmentation methods. (a) Segmentation ground truth. (b) Performance of KMiPC. (c) Performance of KNNiPC. (d) Performance of 3DNCut. (e) Performance of MinCut. (f) Performance of the proposed McPM segmentation.

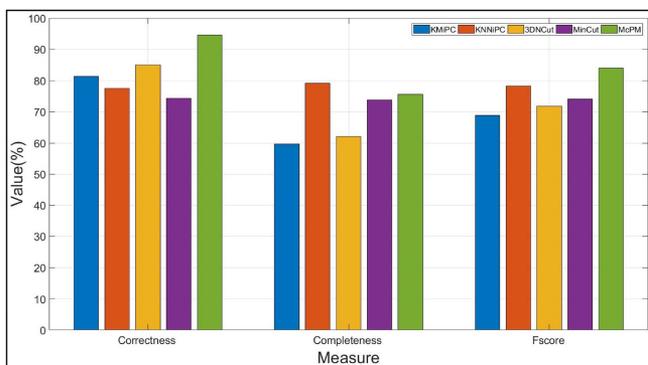


Fig. 16. Evaluation of different segmentation methods.

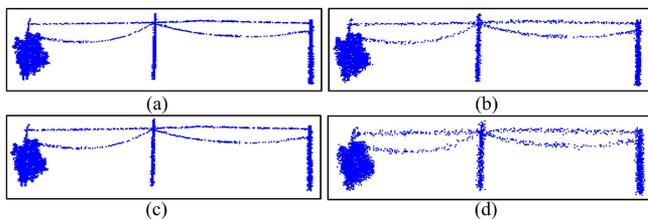


Fig. 17. Experimental scene with Gaussian noises. (a) Gaussian noise  $N(0, 0.01)$ . (b) Gaussian noise  $N(0, 0.02)$ . (c) Gaussian noise  $N(0, 0.05)$ . (d) Gaussian noise  $N(0, 0.1)$ .

and the optimal label configuration is achieved by solving the McPM in graph  $G$ .

*Why do we need the smoothness term in the energy function?* In the energy function formulation, the data term relies on the principal direction and normal vector, which will group neighboring spans into one component. Therefore, we use a smoothness term to constrain the consistency of neighbor points' labels based on the Euclidean distance. Our smoothness term helps adjust points' wrong labels based on its neighboring labels. Having considered that MLS data are noisy and uneven, the smoothness term is formulated by the robust estimator. The robustness of the proposed estimator will be evaluated in Section VII-D.

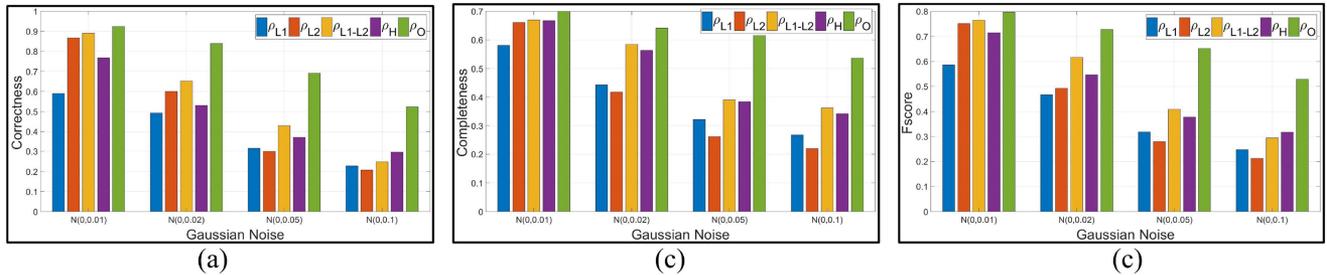


Fig. 18. Comparison of our segmentation using different estimators. (a) Correctness of different estimators. (b) Completeness of different estimators. (c) Fscore of different estimators.

### C. Comparison of Different Segmentation Methods

This section evaluates the segmentation performance of KMiPC [11], KNNiPC [12], 3DNCut [13], MinCut [14], and the proposed McPM in Fig. 15. Fig. 15(a) shows the segmentation ground truth. Fig. 15(b)–(f) shows the performance of KMiPC, KNNiPC, 3DNCut, MinCut, and the proposed McPM, respectively. In the visualization, we use different colors to distinguish components.

KMiPC and 3DNCut segment objects depend on the Euclidean distances between points and they require initializing the number of labels for each scene. The shortcoming is that they fail to detect overlapping points between different objects. Although KNNiPC and MinCut achieve electric poles successfully, they assign spatially close power lines with the same label. In the performance of McPM, both poles and power lines are segmented effectively. Although the segmentation of overlapping regions between vegetations and poles are worse than MinCut, we succeed to provide accurate line components for the subsequent individual span extraction.

The quantitative evaluation method is similar to (12). The correctness measures the ratio of correctly segmented objects in the result and completeness measures the percentage of correctly segmented objects in the reference. Fscore is the harmonic mean of correctness and completeness. The comparison results are shown in Fig. 16, which shows the superiority of our accuracy to other methods.

### D. Evaluation of the Proposed Robust Estimator

To evaluate the performance of our robust estimator, we add different levels of Gaussian noises  $N(\mu, \delta^2)$  to our experimental scene.  $\mu$  is 0 and  $\delta^2$  is selected from  $\{0.01, 0.02, 0.05, 0.1\}$ . Points added with different levels of Gaussian noise are shown in Fig. 17.

The evaluation of different estimators is shown in Fig. 18. We test our segmentation performance using different estimators. Performances of estimators are close to each other if  $\delta^2$  is small, e.g., in the case of  $N(0, 0.01)$ . When we increase the value of  $\delta^2$ , the performance of all estimators will fall down. One can find that the proposed estimator  $\rho_O$  is more accurate and robust than other existing estimators in all cases.

## VIII. CONCLUSION

This paper investigates a new method for power line extraction from mobile LiDAR point clouds. The extraction of power

lines is from components obtained from the optimal segmentation. Our segmentation does not require the initial number of targets and achieves accurate segmentation without human-computer intersection. The main advantage of the proposed extraction is that power lines are extracted from components rather than from points. Each component belongs to one object only, which is easy for the extraction process based on the line property information. In order to deal with the Gaussian noise generated in the data collection, we propose a new robust estimator to improve the robustness of the segmentation. Experiments show that the proposed power line extraction works effectively for MLS data, and the line merging process is able to extract individual spans accurately.

In the future, we will pay attention to the adaptive extraction of power lines for dealing power lines with the heavy sag. Besides, we also expect to explore the power line extraction in the forest environment.

## REFERENCES

- [1] R. V. Ussyshkin, L. Theriault, M. Sitar, and T. Kou, "Advantages of airborne LIDAR technology in power line asset management," in *Proc. Int. Workshop Multi-Platform/Multi-Sens. Remote Sens. Mapping*, 2011, pp. 1–5.
- [2] R. A. McLaughlin, "Extracting transmission lines from airborne LIDAR data," *IEEE Geosci. Remote Sens. Lett.*, vol. 3, no. 2, pp. 222–226, Apr. 2006.
- [3] H. B. Kim and G. Sohn, "Point-based classification of power line corridor scene using random forests," *Photogrammetric Eng. Remote Sens.*, vol. 79, no. 9, pp. 821–833, 2013.
- [4] B. Guo, X. Huang, F. Zhang, and G. Sohn, "Classification of airborne laser scanning data using jointboost," *ISPRS J. Photogrammetry Remote Sens.*, vol. 100, pp. 71–83, 2015.
- [5] B. Guo, Q. Li, X. Huang, and C. Wang, "An improved method for power-line reconstruction from point cloud data," *Remote Sens.*, vol. 8, no. 1, p. 36, 2016.
- [6] J. Liang, J. Zhang, K. Deng, Z. Liu, and Q. Shi, "A new power-line extraction method based on airborne LIDAR point cloud data," in *Proc. Int. Symp. Image Data Fusion*, 2011, pp. 1–4.
- [7] Y. Jwa and G. Sohn, "A piecewise catenary curve model growing for 3D power line reconstruction," *Photogrammetric Eng. Remote Sens.*, vol. 78, no. 12, pp. 1227–1240, 2012.
- [8] L. Zhu and J. Hyypää, "Fully-automated power line extraction from airborne laser scanning point clouds in forest areas," *Remote Sens.*, vol. 6, no. 11, pp. 11267–11282, 2014.
- [9] H. Guan, Y. Yu, J. Li, Z. Ji, and Q. Zhang, "Extraction of power-transmission lines from vehicle-borne LIDAR data," *Int. J. Remote Sens.*, vol. 37, no. 1, pp. 229–247, 2016.
- [10] L. Cheng, L. Tong, Y. Wang, and M. Li, "Extraction of urban power lines from vehicle-borne LIDAR data," *Remote Sens.*, vol. 6, no. 4, pp. 3320–3320, 2014.
- [11] G. Lavoué, F. Dupont, and A. Baskurt, "A new CAD mesh segmentation method, based on curvature tensor analysis," *Comput.-Aided Des.*, vol. 37, no. 10, pp. 975–987, 2005.

- [12] K. Klasing, D. Wollherr, and M. Buss, "A clustering method for efficient segmentation of 3D laser data," in *Proc. Int. Conf. Robot. Automat.*, 2008, pp. 4043–4048.
- [13] Y. Yu, J. Li, H. Guan, C. Wang, and J. Yu, "Semiautomated extraction of street light poles from mobile LIDAR point-clouds," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 3, pp. 1374–1386, Mar. 2015.
- [14] A. Golovinskiy, V. G. Kim, and T. Funkhouser, "Shape-based recognition of 3D point clouds in urban environments," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, 2009, pp. 2154–2161.
- [15] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.
- [16] Y. Boykov and G. Funka-Lea, "Graph cuts and efficient ND image segmentation," *Int. J. Comput. Vis.*, vol. 70, no. 2, pp. 109–131, 2006.
- [17] D. Munoz, N. Vandapel, and M. Hebert, "Directional associative Markov network for 3-D point cloud classification," in *Proc. Int. Symp. 3D Data Proc., Vis. Transmiss.*, 2008, pp. 65–72.
- [18] F. R. Hampel, E. M. Ronchetti, P. J. Rousseeuw, and W. A. Stahel, *Robust Statistics: The Approach Based on Influence Functions*, vol. 114. Hoboken, NJ, USA: Wiley, 2011.
- [19] P. J. Huber, *Robust Statistics*. Berlin, Germany: Springer, 2011.
- [20] O. Veksler and R. Zabih, *Efficient Graph-Based Energy Minimization Methods in Computer Vision*. Ithaca, NY, USA: Cornell Univ. Press, 1999.
- [21] S. Xu, R. Wang, and H. Zheng, "An optimal hierarchical clustering approach to segmentation of mobile LIDAR point clouds," 2017, arXiv:1703.02150.
- [22] S. Xia and R. Wang, "A fast edge extraction method for mobile LIDAR point clouds," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 8, pp. 1288–1292, Aug. 2017.
- [23] C. Chen, B. Yang, S. Song, X. Peng, and R. Huang, "Automatic clearance anomaly detection for transmission line corridors utilizing UAV-borne LIDAR data," *Remote Sens.*, vol. 10, no. 4, p. 613, 2018.
- [24] J. Munkres, "Algorithms for the assignment and transportation problems," *J. Soc. Ind. Appl. Math.*, vol. 5, no. 1, pp. 32–38, 1957.



**Sheng Xu** received the B.Eng. degree in computer science and technology from Nanjing Forestry University, Nanjing, China, and the Ph.D. degree in digital imaging systems from the University of Calgary, Calgary, AB, Canada, in 2010 and 2018, respectively.

He is currently an Assistant Professor with the College of Information Science and Technology, Nanjing Forestry University, which he joined in 2018. His current research interests include laser scanning, vegetation mapping, and computer vision.



**Ruisheng Wang** received the B.Eng. degree in photogrammetry and remote sensing from Wuhan University, Wuhan, China, the M.Sc.E. degree in geomatics engineering from the University of New Brunswick, Saint John, NB, Canada, and the Ph.D. degree in electrical and computer engineering from McGill University, Montreal, QC, Canada, in 1993, 2004 and 2011, respectively.

He is currently an Associate Professor with the Department of Geomatics Engineering, University of Calgary, Calgary, AB, Canada, which he joined in 2012. Prior to that, he worked as an Industrial Researcher with Nokia, Chicago, IL, USA, since 2008. His research interests include geomatics and computer vision.