IET Intelligent Transport Systems

Research Article

IET Journals
The Institution of Engineering and Technology

Automated traffic sign and light pole detection in mobile LiDAR scanning data

ISSN 1751-956X
Received on 20th July 2018
Revised 13th November 2018
Accepted on 11th December 2018
E-First on 14th January 2019
doi: 10.1049/iet-its.2018.5360
www.ietdl.org

Mohammadreza Javanmardi[1], *Ziqi Song*[2] ✉, *Xiaojun Qi*[1]

[1]*Department of Computer Science, Utah State University, Logan, UT 84322-4205, USA*
[2]*Department of Civil and Environmental Engineering, Utah State University, Logan, UT 84322-4110, USA*
✉ *E-mail: ziqi.song@usu.edu*

Abstract: Detection of traffic signs and light poles using light detection and ranging (LiDAR) data has demonstrated a valid contribution to road safety improvements. In this study, the authors propose a fast and reliable method, which can identify various traffic signs and light poles in mobile LiDAR data. Specifically, they first use the surface reconstruction algorithm to extract the normal vectors of the points as one of the characteristic features and apply *k*-means on the characteristic features of the points to automatically segment the data into road or non-road points. They then employ sliding cuboids to search for high-elevated objects that are located near the borders and on top of the road points. They further employ the random sample consensus algorithm to remove outliers and keep the points that fall on the perpendicular planes to the road trajectory. Finally, they introduce a modified seeded region growing algorithm to remove noisy points and incorporate the shape information to reject the false objects. A set of extensive experiments have been carried out on the datasets that are captured by Utah Department of Transportation from I-15 highway. The results demonstrate the robustness of the proposed method in detecting almost all traffic signs and light poles.

1 Introduction

Mobile light detection and ranging scanning (MLS) has been considered as one of the most growing technologies in recent years and has been utilised in various applications such as digital terrain models, three-dimensional (3D) asset inventory mapping, streetscape design, and traffic sign extraction. Significant effort has been made in recent years to automatically segment and classify large-scale MLS datasets into different regions. The resultant regions of interest (ROIs) vary from one study to another. However, a majority of the methods categorise the datasets into specific classes such as cars, humans, light poles, traffic signs, and so on. Here, we briefly review several representative approaches to processing MLS datasets.

Several methods project 3D information onto the 2D grid to reduce the complexity and the computational time. The 2D grid is called the elevation image or the digital elevation model and each pixel in the 2D grid contains the elevation information. Hernández and Marcotegui [1] introduce a method to project the data to elevation images and then propose a segmentation method based on morphological operations and support vector machines (SVMs) to classify the data into four categories, i.e. car, lamp post, pedestrian, and others. Similarly, Zhu *et al.* [2] initially project MLS data to a range image, in which columns represent the sequential order of measurement, rows represent the acquisition time of each laser scan line, and pixel values code the distance from the sensor to the point. They then use SVMs and the decision tree-based segmentation-classification method on the projected points to segment their dataset into regions of different objects. Projecting the MLS data onto the elevation images and performing segmentation on the projected images is fast and can be mainly used for guiding autonomous vehicles, which does not require high accuracy but requiring fast speed to detect and predict obstacles in real time.

The slower but more accurate methods directly process the 3D point clouds. These approaches are more applicable for the cases that high accuracy is valued more than fast speed. For instance, the task of traffic sign maintenance and road inspection requires an accurate detection of the locations of the road. Douillard *et al.* [3] propose a set of segmentation methods for 3D data based on voxelisation and meshing. They also provide empirical evidence of the benefit of extracting ground as the prior for the segmentation method to improve the accuracy. Lin *et al.* [4] propose a statistics-based approach using principal component analysis (PCA) and geometric median to extract eigen-features from a local set of point clouds and segment the eigen-feature-based data into building and non-building categories.

Since traffic signs are one of the most important road features, various mature traffic sign extraction methods have been proposed for images and videos. For example, Soheilian *et al.* [5] present an automatic approach for utilising the colour information to identify the silhouette of signs in every individual image. They then propose a multi-view constrained 3D reconstruction algorithm to provide an optimum 3D silhouette to detect traffic signs. Adam and Ioannidis [6] propose to extract traffic signs by using colour images acquired by a camera mounted on the moving vehicle. They detect the ROIs and classify them as traffic or non-traffic signs by feeding the regions' histogram of oriented gradient (HOG) descriptors to a trained SVM. Khalid *et al.* [7] estimate a global threshold value using the correlation property of a given image and segment the regions of traffic signs based on the global threshold and morphological operations. They further detect the traffic signs by feeding HOG descriptors to a trained SVM-*k*-nearest neighbour classifier.

Recently, researchers have proposed various methods to assist the implementation of mobile laser technology for safer navigation of intelligent vehicles and better road maintenance, inspection, and safety. However, few state-of-the-art methods extensively study the light detection and ranging (LiDAR) data. Pu *et al.* [8] introduce one of the pioneer studies in detecting and distinguishing the traffic signs using LiDAR data. In this work, they initially segment the data into one of the three coarse categories including the ground surface, the objects on the ground, and the objects off the ground. They further use the size, shape, and orientation information to classify the on-ground points to more detailed classes such as traffic signs. Yokoyama *et al.* [9] employ PCA to extract the pole-like objects from MLS data and classify them into utility poles, lamp posts, or street signs. Yu *et al.* [10] propose a voxel-based upward growing method to remove the ground points and a voxel-based normalised cut to segment the remaining MLS point clouds

data into street light poles, traffic signposts, or bus stations. Riveiro *et al.* [11] employ the geometric and radiometric information of retro-reflective traffic signs in the segmentation process to compute the optimal intensity threshold to separate the traffic signs from the backgrounds. Lehtomaki *et al.* [12] propose to remove ground and building points from the original data to reduce the search space. They then incorporate three sets of features, i.e. local descriptor histogram, spin images, and general shape and point distribution, to segment the data into trees, lamp posts, traffic signs, cars, penetrations, and hoardings.

In this paper, we propose an effective method to automatically detect traffic signs and light poles from MLS point clouds in highway areas without any pre-processing or learning steps. Specifically, we first extract the road points (ground points) from the original dataset without involving any manual pre-processing steps. We then detect traffic sign and light pole candidates near the border or on top of the roads and automatically classify them to the corresponding classes without involving any learning processes. Finally, we remove false objects by using shape information of traffic sign and light pole candidates. Unlike the proposed method, previous studies [8–11] mainly extract traffic signs and light poles in urban areas, where either manual pre-processing is required to remove the road points (ground points) [9] or a training process is employed to learn a classifier for identifying the objects [12]. Their performance is significantly reduced when processing the mobile LiDAR data in highway areas due to the sloping roads and the existence of highway objects such as billboards, bridges, high elevated curbs, trees, and so on. In addition, some of the existing methods are mostly based on pole-like features [8, 9, 13], which are not effective to deal with light poles attached with traffic signs or advertising boards and light poles near trees. The contributions of this paper are as follows:

- Using the surface reconstruction algorithm to extract normal vectors of the points as one of the selected characteristic features;
- Proposing an unsupervised road point extraction scheme by applying the *k*-means clustering on the characteristic features;
- Designing a sliding cuboid to identify groups of candidate points by searching for the high elevated objects above or beside the roads;
- Employing the random sample consensus (RANSAC) algorithm in a novel and unique way to select the robust candidate points by removing high elevated outliers that do not represent perpendicular planes along the vehicle trajectory;
- Proposing a LiDAR modified seeded region growing algorithm to remove the noisy points around the objects;
- Introducing a two-step post-processing method to remove false positive objects.

The remaining sections of the paper are organised as follows. Section 2 presents the proposed method for extracting road, traffic signs, and light poles. Section 3 presents the experimental results on 8 mileposts of the I-15 highway and evaluates the performance of the proposed method and its variant method without post-processing steps. Section 4 draws the conclusion and presents the direction of future work.

2 Proposed method

The proposed method consists of three main components, namely, road point extraction, traffic sign and light pole extraction and classification, and post-processing. For the road point extraction component, we utilise the surface reconstruction algorithm to extract the characteristic features such as altitude, reflectivity, and orientation to represent each MLS point and apply the unsupervised *k*-means on the characteristic features to quickly extract candidate road points. For the traffic sign and light pole extraction and classification component, we design a sliding cuboid to quickly identify groups of candidate points along the road points and employ RANSAC to remove non-planar false candidate points along the vehicle trajectory. This step not only removes the non-planar objects but also eliminates the outliers from the LiDAR dataset. For the post-processing component, we utilise the modified seeded region growing to remove the outlier points around the candidates and employ shape information to remove the false objects. By employing RANSAC and modified seeded region growing techniques, we are able to successfully extract both traffic signs and light poles in the noisy dataset. We believe that other de-noising techniques [14, 15], with some modifications, may yield similar performance as the proposed method. In the following, we explain each component in detail.

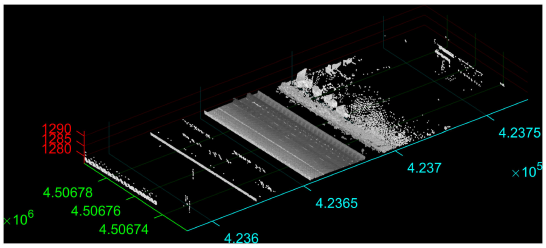

Fig. 1 *Demonstration of a section of the road on top of the global Cartesian coordinate system, where x-axis is shown in cyan, y-axis is shown in green, and z-axis is shown in red*

2.1 Road point extraction

The original point clouds, as the input, contain a large number of 4D points. Each point includes the global positional values (i.e. x, y, and z) and the intensity value. The intensity is a measure of the returned strength of the laser pulse that is generated from the point. To reduce the computational time and facilitate processing, we segment the original point clouds to a predefined number of sections. Since the elevation of the points may gradually vary along the uneven or hilly road direction, the points in a smaller section tend to have more similar elevation than the points in a larger section. Each section is a portion of the dataset along the length of the road (i.e. y-axis). In order to illustrate the procedure of vertical section construction, we demonstrate a section of the road in the global Cartesian coordinate system as shown in Fig. 1, where the x, y, and z axes are highlighted in cyan, green, and red, respectively. The advantage of using the global Cartesian coordinate system is that the direction of x-axis and y-axis is fixed even when the road rotates and does not keep straight. We then process each section separately in the following steps and the resultant points are concatenated to uniquely represent the road.

2.1.1 Normal vector estimation: We aim to estimate the normal vector for each data point to represent its orientation. We adopt the surface construction method [16], which uses a fixed number of neighbouring points to fit a local plane to determine the normal vector of each data point in a section. For each section l, we include the location information (e.g. x, y, and z) of all the points in a set of 3D data, i.e. $\mathbf{P}_l = \{\boldsymbol{p}_{l,1}, \ldots, \boldsymbol{p}_{l,n_l}\} \subset \mathbb{R}^3$, where n_l is the number of data points in section l. We first find six nearest neighbours of the point $\boldsymbol{p}_{l,i}$ (e.g. $nbhd(\boldsymbol{p}_{l,i})$) to fit a local plane and compute the normal vector $\bar{\boldsymbol{n}}_{l,i}$ associated with each point $\boldsymbol{p}_{l,i}$ in set $\mathbf{P}_l$. We then compute the centroid of the points in $nbhd(\boldsymbol{p}_{l,i})$ as the centre $\bar{\boldsymbol{p}}_{l,i}$. We finally employ the PCA method on $nbhd(\boldsymbol{p}_{l,i})$ to calculate the smallest principal vector as the normal vector $\bar{\boldsymbol{n}}_{l,i}$. To do so, we form the 3×3 covariance matrix $\boldsymbol{C}$ of $nbhd(\boldsymbol{p}_{l,i})$ as follows:

$$\boldsymbol{C} = \sum_{\boldsymbol{p} \in nbhd(\boldsymbol{p}_{l,i})} (\boldsymbol{p} - \bar{\boldsymbol{p}}_{l,i}) \otimes (\boldsymbol{p} - \bar{\boldsymbol{p}}_{l,i})^{\top} \tag{1}$$

where $\otimes$ denotes the outer product operator and $\boldsymbol{C}$ is symmetric positive semi-definite. We then calculate the normal vector $\bar{\boldsymbol{n}}_{l,i}$ of the point $\boldsymbol{p}_{l,i}$ by solving the following optimisation function:

$$\underset{\bar{\boldsymbol{n}}_{l,i}}{\text{minimise}} \quad \bar{\boldsymbol{n}}_{l,i}^{\top} \boldsymbol{C} \bar{\boldsymbol{n}}_{l,i} \tag{2a}$$

$$\text{subjectto} \quad \bar{\boldsymbol{n}}_{l,i}^{\top}\bar{\boldsymbol{n}}_{l,i} = 1 \tag{2b}$$

This optimisation problem can be solved by introducing the Lagrange multiplier γ and setting the Lagrangian derivative to zero. The following equation can be derived:

$$\det(\boldsymbol{C} + \boldsymbol{C}^{\top} - 2\gamma\boldsymbol{I}) = 0 \tag{3}$$

Here, $\boldsymbol{I}$ is the identity matrix in $\mathbb{R}^3$. We can infer that γ and $\bar{\boldsymbol{n}}_{l,i}$ are eigenvalue and eigenvector of the covariance matrix $\boldsymbol{C}$, respectively. Let the eigenvalues of $\boldsymbol{C}$ be λ_i^1, λ_i^2, and λ_i^3 (i.e. $\lambda_i^1 \geq \lambda_i^2 \geq \lambda_i^3$) and their associated unit eigenvectors be $\hat{\boldsymbol{v}}_i^1, \hat{\boldsymbol{v}}_i^2$, and $\hat{\boldsymbol{v}}_i^3$, respectively. It is proven that γ is the smallest eigenvalue (i.e. λ_i^3) and $\bar{\boldsymbol{n}}_{l,i}$ is either $\hat{\boldsymbol{v}}_i^3$ or $-\hat{\boldsymbol{v}}_i^3$ such that

$$\boldsymbol{C}\bar{\boldsymbol{n}}_{l,i} = \gamma\bar{\boldsymbol{n}}_{l,i} \tag{4}$$

Multiplying $\bar{\boldsymbol{n}}_{l,i}$ on both sides of (4), we obtain

$$\bar{\boldsymbol{n}}_{l,i}^{\top}\boldsymbol{C}\bar{\boldsymbol{n}}_{l,i} = \gamma\bar{\boldsymbol{n}}_{l,i}^{\top}\bar{\boldsymbol{n}}_{l,i} \tag{5}$$

Since $\bar{\boldsymbol{n}}_{l,i}^{\top}\bar{\boldsymbol{n}}_{l,i} = 1$, (5) can be simplified to

$$\bar{\boldsymbol{n}}_{l,i}^{\top}\boldsymbol{C}\bar{\boldsymbol{n}}_{l,i} = \gamma \tag{6}$$

The left side of (6) is the same optimisation function in (2). Therefore, γ is the smallest eigenvalue and its eigenvector is $\bar{\boldsymbol{n}}_{l,i}$. Here, we denote $\bar{\boldsymbol{n}}_{l,i}$ as $\bar{\boldsymbol{n}}_{l,i} = u\hat{i} + v\hat{j} + w\hat{k}$, where u, v, and w are the projected components in the global Cartesian coordinate system.

2.1.2 Data points clustering: We aim to cluster the data points to road or non-road classes by incorporating the normal vector of the points as one of the characteristic features. Specifically, for each data point in the point clouds, we propose to concatenate its 3D normal vector (i.e. orientation) with its z value (i.e. elevation) and intensity value (i.e. reflectivity) to construct a 5D feature vector to represent its road characteristics. We denote the set of the new characteristic features of the data points in section l as $\mathbf{P}_l' = \{\boldsymbol{p}_{l,1}', \ldots, \boldsymbol{p}_{l,n_l}'\} \subset \mathbb{R}^5$. We then employ the k-means clustering algorithm [17], a powerful unsupervised method with k being 2, on $\mathbf{P}_l'$ to group all the data points into either the road or non-road cluster. Since the MLS point clouds mostly contain the road points, we select the cluster with the larger number of points as the road cluster. The road clusters within each section, $\mathbf{Road}_l$, are then concatenated to contain the extracted road points in a set denoted as $\mathbf{Road}_{\text{All}}$. Algorithm 1 (see Fig. 2) summarises the proposed method to extract $\mathbf{Road}_{\text{All}}$.

2.2 Traffic sign and light pole extraction and classification

Traffic signs are mostly located near the border or on top of the roads to be visible for drivers. In addition, light poles are raised sources of light on the edges of a road or path. We use this prior information to eliminate the data points that are located off the road and eliminate the off-road counterfeit objects such as billboards and buildings that might have similar characteristics such as intensity and elevation as the traffic signs or light poles. As a result, we effectively reduce the search space in high-density MLS point clouds and reduce the computational time. In addition, we utilise some observational statistics (e.g. height, elevation, and planar projection of traffic signs or light poles) to further remove the points that are unlikely to be road points.

2.2.1 Cuboid searching: We aim to utilise a sliding rectangular cuboid to find the traffic signs located near the border or on top of the road and the light poles that are situated near the borders of the roads. Specifically, we use the road points in each section l (e.g. $\mathbf{Road}_l$) to find their ranges for the x and y values. We then select the points in $\mathbf{P}_l$, whose x and y values fall into these calculated ranges. The selected set is denoted as $\mathbf{Q}_l = \{\boldsymbol{q}_{l,1}, \ldots, \boldsymbol{q}_{l,m_l}\} \subset \mathbb{R}^4$, where m_l is the number of points in $\mathbf{Q}_l$ and $m_l \ll n_l$ due to the elimination of a significant number of off-road points. It should be noted that the points in $\mathbf{Q}_l$ may correspond to roads, traffic signs, light poles, bridges, moving vehicles, and other objects near or on the roads. To solve this issue, we design a rectangular cuboid search strategy to identify the points in $\mathbf{Q}_l$ that correspond to the traffic signs and light poles.

We first define the starting point in xy-plane of each section l as $S_l = (x_{l,s}, y_{l,s})$, where $x_{l,s}$ and $y_{l,s}$ are the minimum x and y values of the points in $\mathbf{Q}_l$. We then put a rectangular cuboid with the dimension of $4 \times 4 \times inf$ (i.e. no limitation in the z-direction) at S_l and continue moving this sliding cuboid along 4 m at the x-direction or 4 m at the y-direction until the sliding cuboids cover all the points in $\mathbf{Q}_l$. For each sliding cuboid that contains any points in $\mathbf{Q}_l$, we put these points in a set denoted as $\mathbf{W}_l^{(c_i)} = \{\boldsymbol{w}_{l,1}^{(c_i)}, \ldots, \boldsymbol{w}_{l,m_l'}^{(c_i)}\} \subset \mathbb{R}^4$, where c_i is the upper-left corner of the satisfied sliding cuboid and m_l' is the number of points within this cuboid. We observe that a majority of traffic signs have an elevation range of more than 3 m and are located above 1.5 m from the road surface. Moreover, all the light poles have the range of elevation more than 3 m. Therefore, we first utilise the height information to filter out some obvious outliers by removing the $\mathbf{W}_l^{(c_i)}$ set, whose range of z values is <3. For each kept set $\mathbf{W}_l^{(c_i)}$, we further remove any points with the height values less than an adaptive threshold, which is computed by a predefined value (e.g. 1.5) plus the mean height value of all the points in $\mathbf{W}_l^{(c_i)}$. The remaining points in a kept $\mathbf{W}_l^{(c_i)}$ forms the candidate set, $\tilde{\mathbf{X}}_l^{(c_i)} = \{\tilde{\boldsymbol{x}}_{l,1}^{(c_i)}, \ldots, \tilde{\boldsymbol{x}}_{l,m_l''}^{(c_i)}\} \subset \mathbb{R}^4$, where m_l'' is the number of points in the candidate set and $m_l'' < m_l'$.

Input: An unorganized set $\mathbf{P}$ of 4-dimensional MLS point cloud data (x, y, z coordinates and their intensities)
Output: An unorganized set of MLS point cloud data for each section (e.g., $\mathbf{P}_l$), road points in each section (e.g., $\mathbf{Road}_l$), and road points in the MLS point cloud data (e.g., $\mathbf{Road}_{All}$)

1.Segment the input data $\mathbf{P}$ into a predefined number of sections (e.g., $N = 24$ sections) along the y-axis, where the data in each section l ($1 \leq l \leq N$) is denoted as $\mathbf{P}_l$.
2.For each point $\boldsymbol{p}_{l,i}$ in $\mathbf{P}_l$
 (a)Find its normal vector $\bar{\boldsymbol{n}}_{l,i}$ by the following steps:
  (i)Find 6 nearest neighbors of $\boldsymbol{p}_{l,i}$, which are denoted as $nbhd(\boldsymbol{p}_{l,i})$.
  (ii)Compute the covariance matrix $\boldsymbol{C}$ of $nbhd(\boldsymbol{p}_{l,i})$ using Eq. (1).
  (iii)Calculate the smallest eigenvalue λ_i^3.
  (iv)Select the eigenvector $\hat{\boldsymbol{v}}_i^3$ corresponding to λ_i^3 as the normal vector $\bar{\boldsymbol{n}}_{l,i} = u\hat{i} + v\hat{j} + w\hat{k}$.
 (b)Concatenate its normal vector components (i.e., u, v, w) with its z and intensity values to construct a 5-dimensional characteristic feature vector $\boldsymbol{p}_{l,i}'$.
 Endfor
3.For each section l
 (a)Apply the k-means algorithm on $\mathbf{P}_l'$, a set containing the 5 characteristic features of all points in section l, with k being 2.
 (b)Select the cluster with the larger number of points as the road point cluster $\mathbf{Road}_l$.
 Endfor
4.Concatenate $\mathbf{Road}_l$ to form the road points $\mathbf{Road}_{All}$ in the MLS point cloud data.

Fig. 2 *Algorithm 1: road point extraction*

2.2.2 Plane fitting: We aim to utilise the planary property to extract traffic signs and light poles by finding the points that represent planes. Specifically, in order to identify the ROIs in the candidate set $\tilde{\mathbf{X}}_l^{(c_i)}$ for each cuboid location c_i in section l, we adopt the RANSAC algorithm [18] to discard the points that do not belong to a plane estimated by a sufficient number of inliers.

Input: An unorganized set of MLS point cloud data for each section (e.g., $\mathbf{P}_l$), road points in each section (e.g., $\mathbf{Road}_l$), the reference vector of [0 -1 0], and an intensity threshold T

Output: Traffic sign and light pole candidates $\check{\mathbf{X}}_l^{(c_i)}$s.

1.For each section l
  (a)Find the ranges for x and y values of the points in $\mathbf{Road}_l$
  (b)Select the input points in $\mathbf{P}_l$ that fall in these ranges to construct set $\mathbf{Q}_l$.
  (c)Compute the search starting point $S_l = (x_{l,s}, y_{l,s})$, where $x_{l,s}$ and $y_{l,s}$ are the minimum x and y values of the points in $\mathbf{Q}_l$.
  (d)Construct a rectangular cuboid with the dimension of $4 \times 4 \times inf$ at S_l.
  (e)Continue moving this sliding cuboid along 4 meters at the x direction or 4 meters at the y direction until the sliding cuboids cover all the points in $\mathbf{Q}_l$.
  (f)For the location c_i of each sliding cuboid that contains any points in $\mathbf{Q}_l$
    (i)Find the points in $\mathbf{Q}_l$ that fall inside the corresponding sliding cuboid and put these points in set $\mathbf{W}_l^{(c_i)}$.
    (ii)Calculate the range of z value of the points in $\mathbf{W}_l^{(c_i)}$ as Rng_Z.
    (iii)If $Rng_Z \geq 3$
      (A)Calculate the minimum z value of all the points in $\mathbf{W}_l^{(c_i)}$ as min_z
      (B)For each point $\boldsymbol{w}_{l,j}^{(c_i)}$ (i.e., (x_w, y_w, z_w, int_w)) in $\mathbf{W}_l^{(c_i)}$
        •If $z_w > min_z + 1.5$
          ○Add $\boldsymbol{w}_{l,j}^{(c_i)}$ to the candidate set $\check{\mathbf{X}}_l^{(c_i)}$.
        Endif
      Endfor
      (C)Use RANSAC to remove any non-planar point $\check{\boldsymbol{x}}_{l,k}^{(c_i)} = (x_{\check{x}}, y_{\check{x}}, z_{\check{x}}, int_{\check{x}}) \in \check{\mathbf{X}}_l^{(c_i)}$.
    Endif
    (iv)If $mean(int_{\check{x}}) < T$ for all the points $\check{\boldsymbol{x}}_{l,k}^{(c_i)}s \in \check{\mathbf{X}}_l^{(c_i)}$
      •Discard the candidate set $\check{\mathbf{X}}_l^{(c_i)}$
    Endif
  Endfor
  (g)Keep all $\check{\mathbf{X}}_l^{(c_i)}$ in section l as candidates.
Endfor
2.Keep candidates $\check{\mathbf{X}}_l^{(c_i)}$ for all sections as the final traffic sign and light pole candidates.

Fig. 3 *Algorithm 2: traffic sign and light pole extraction*

RANSAC is an iterative method for estimating parameters of a mathematical model from a set of data points containing outliers. The input to RANSAC contains the candidate set $\check{\mathbf{X}}_l^{(c_i)}$, a parametrised fitting model, and two confidence parameters (e.g. the maximum distance and the maximum angular distance). The parametrised model is a reference normal vector, whose element is the projected value at the *y*-direction. This vector is used as an orientation constraint to fit a plane that has an approximate normal vector similar to the reference vector. RANSAC fits a plane to the input points to achieve the maximum distance and the maximum angular distance from the inlier points to the plane.

For a candidate set $\check{\mathbf{X}}_l^{(c_i)}$, RANSAC iteratively selects a random subset from this set and tests a hypothesis that the points in the selected subset are inliers. A plane is fitted to the hypothetical inliers and all other data points are then examined against the fitted plane. The points fitting well to the estimated model are considered as hypothetical inliers. The plane is iteratively reestimated from the inliers. If a sufficient number of points are evaluated as inliers, the estimated plane is considered as reasonably good. Otherwise, it is rejected. The candidate plane is finally evaluated by estimating the error of the inliers to the plane. This process is repeated for a fixed number of times. We use the resultant inlier indices to select the points within $\check{\mathbf{X}}_l^{(c_i)}$ as the candidates for traffic signs and light poles. In other words, we discard the outlier points in $\check{\mathbf{X}}_l^{(c_i)}$ that do not represent a perpendicular surface to the road direction.

One of the advantages of RANSAC is its robust estimation of plane fitting. Particularly, it can estimate the parameters of planes with a high degree of accuracy even when there are a significant number of outliers in the input data. Its shortcoming is that the results may not be optimal when the number of iterations is limited. As a result, there is a trade-off between the number of iterations and a reasonable output model. In the proposed method, we set the number of iterations to be 1000 to yield overall best output models.

2.2.3 Intensity thresholding: We aim to use the highly reflective property to remove the points that do not belong to traffic signs and light poles. Since traffic signs are covered by highly reflective materials to make them visible during any weather conditions at days and nights and light poles are manufactured with metals, their corresponding point clouds normally have higher intensities. We can then use the intensity information to extract traffic signs and light poles. In the proposed method, we discard a candidate set $\check{\mathbf{X}}_l^{(c_i)}$ if the average intensity of the points within this set is less than a predefined threshold. All the remaining candidate sets $\check{\mathbf{X}}_l^{(c_i)}$ s for all sections ($l = 1, \ldots, 24$) are kept as candidates for traffic signs and light poles. Algorithm 2 (see Fig. 3) summarises the proposed method to extract $\check{\mathbf{X}}_l^{(c_i)}$ s.

2.2.4 Classification: We aim to use the height property to classify the data into traffic sign and light pole classes. Since traffic signs in highway areas usually have lower elevation (height) than light poles, we use this prior information to estimate a threshold for the height of the candidates to automatically classify each $\check{\mathbf{X}}_l^{(c_i)}$ as either the traffic sign or the light pole. For each $\check{\mathbf{X}}_l^{(c_i)}$ set, we first find the range of z of its points. Second, we use an estimated threshold θ to segment the candidate points into two groups: $\mathbf{GR}_1$ consisting of all candidate points whose z ranges are less than θ and $\mathbf{GR}_2$ consisting of remaining candidate points. Third, we calculate the average z ranges μ_1 and μ_2 for the candidate points in groups $\mathbf{GR}_1$ and $\mathbf{GR}_2$, respectively. Fourth, we repeat the process by segmenting the candidate points into two groups using the new $\theta = (\mu_1 + \mu_2)/2$ until the difference between the θ values in two iterations is smaller than a predefined threshold.

Since a dataset might contain traffic signs or light poles or none, we set the final threshold $\Theta = \max(\theta, 15)$ to avoid misclassification. Specifically, we classify the candidates as traffic signs if the range of z of the points is less than Θ. Otherwise, we classify the candidates as light poles. We use $\mathbf{TS}_l^{(c_i)}$ to denote traffic signs in $\check{\mathbf{X}}_l^{(c_i)}$ and use $\mathbf{LP}_l^{(c_i)}$ to denote light poles in $\check{\mathbf{X}}_l^{(c_i)}$. The $\mathbf{TS}_l^{(c_i)}$'s for $l = 1, \ldots, 24$ are concatenated to obtain traffic sign points in the whole dataset as $\mathbf{TS}_{All}$. Similarly, the $\mathbf{LP}_l^{(c_i)}$'s for $l = 1, \ldots, 24$ are concatenated to obtain light pole points in the whole dataset as $\mathbf{LP}_{All}$. Algorithm 3 (see Fig. 4) summarises the proposed method to classify the candidates as traffic sign and light pole classes.

It should be noted that this classification is an unsupervised pattern recognition technique for LiDAR data, which utilises shape information (height) to extract patterns of traffic sign and light pole candidates for classification. The supervised deep learning-based techniques have also been extensively used in object classification tasks [19–23]. However, they require a large amount of clean data for training and validation. For instance, researchers can build various pre-trained networks such as VGG-net [24], AlexNet [19], and ResNet [20] using a large amount of camera images from the web or the other benchmark datasets including ImageNet [25], PASCAL VOC [26], and so on. Moreover, deep learning approaches tend to achieve better performance when there are a sufficient and balanced number of positive and negative samples with the same size. It should be emphasised that deep learning approaches may not necessarily improve the classification accuracy in our task since we do not have a large training dataset containing a sufficient and balanced number of positive samples (e.g. traffic signs and light poles) and negative samples (e.g. billboards, buildings etc.). In addition, we cannot use the available pre-trained 3D object networks (e.g. PointNet [21]) to achieve decent classification accuracy for 3D point cloud data in our case due to distinct distributions and nature between our dataset and the ShapeNetPart dataset [27] that is used to train PointNet. Re-sampling positive and negative data to the same dimension may also lead to some information loss.

Input: Candidates of traffic signs and light poles $\check{\mathbf{X}}_l^{(c_i)}$s.
Output: Traffic sign class ($\mathbf{TS}_{All}$) and light pole class ($\mathbf{LP}_{All}$)

1.For each $\check{\mathbf{X}}_l^{(c_i)} = \{\check{\boldsymbol{x}}_{l,1}^{(c_i)}, \cdots, \check{\boldsymbol{x}}_{l,K_l}^{(c_i)}\}$
  (a)Calculate the range of z values of the points $\check{\boldsymbol{x}}_{l,k}$ for $k = 1 : K_l$
  (b)Add the calculated range to the array $\boldsymbol{R}$
  Endfor
2.Set the initial threshold θ as any value within the range in $\boldsymbol{R}$.
3.For a candidate $\check{\mathbf{X}}_l^{(c_i)}$
  •If its z range in $\boldsymbol{R} < \theta$
    ○Add $\check{\mathbf{X}}_l^{(c_i)}$ to $\mathbf{GR}_1$
  •Else
    ○Add $\check{\mathbf{X}}_l^{(c_i)}$ to $\mathbf{GR}_2$
  Endif
  Endfor
4.Calculate the average z ranges μ_1 and μ_2 for the candidates in the groups $\mathbf{GR}_1$ and $\mathbf{GR}_2$, respectively.
5.Set $\theta = \frac{(\mu_1+\mu_2)}{2}$.
6.Repeat step 3 to 5 until the difference between the θ value in two successive iterations is smaller than a predefined threshold (e.g., 0.5).
7.Set $\Theta = \max(\theta, 15)$ to avoid misclassification in the datasets with traffic signs, light poles, or none.
8.For a candidate $\check{\mathbf{X}}_l^{(c_i)}$
  •If its z range in $\boldsymbol{R} < \Theta$
    ○Add $\check{\mathbf{X}}_l^{(c_i)}$ to $\mathbf{TS}_l^{(c_i)}$
  •Else
    ○Add $\check{\mathbf{X}}_l^{(c_i)}$ to $\mathbf{LP}_l^{(c_i)}$
  Endif
  Endfor
9.Concatenate the $\mathbf{TS}_l^{(c_i)}$s for $l = 1, \cdots, 24$ to obtain $\mathbf{TS}_{All}$.
10.Concatenate the $\mathbf{LP}_l^{(c_i)}$s for $l = 1, \cdots, 24$ to obtain $\mathbf{LP}_{All}$.

Fig. 4 *Algorithm 3: traffic sign and light pole classification*

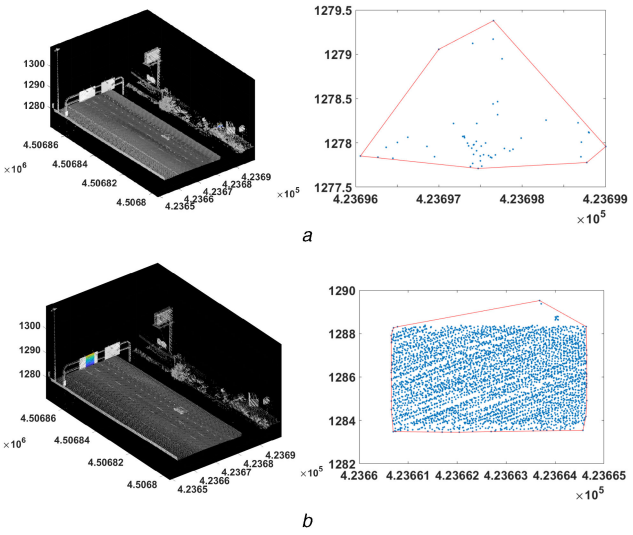

Fig. 5 *Illustration of the results obtained by Algorithm 2 (Fig. 3) for two sections*
(a) Region that represents a part of the tree and its projection on the *xz*-plane together with its convex hull shown in red, ***(b)*** Region that represents a traffic sign and its projection on the *xz*-plane together with its convex hull shown in red

2.3 Post-Processing

We aim to utilise shape information to eliminate the false objects from the classification results. To this end, we propose a two-step post-processing method to achieve more reliable performance by removing the outliers of the projection points in *xz*-plane and taking the shape of the cleaned projection points into consideration.

We observe that the *xz* projection of the points representing traffic signs and light poles are approximately rectangular shapes. The projection of the points that belong to false objects (e.g. trees, bridges, buildings, and vehicles on the road) usually does not exhibit the rectangular shape. For example, the top row of Fig. 5 presents two kinds of regions obtained by Algorithm 2 (Fig. 3) for a section. It is clear that the convex hull around the projected points of a traffic sign is more likely to be a rectangle than the convex hull around the projected points of a tree. We propose to use the shape information of the objects to distinguish the ROIs from the false objects.

Since outlier points can be randomly located around each object, they may make the overall convex hull of the projected points to be non-rectangular. To address this issue, we propose the modified seeded region growing algorithm to eliminate the outliers around the object. We then use the rectangular shape information to eliminate the false objects.

2.3.1 Modified seeded region growing: The seeded region growing algorithm is conventionally considered as a region-based segmentation method. It is a bottom-up procedure that starts with a set of seed pixels in an image. The goal is to grow a uniform and connected region from each seed. This method is extensively used as a segmentation process so analysis of an image can be automatically performed on the segmentation results.

In this work, we propose the modified seeded region growing algorithm on the LiDAR data to group the potential object points and eliminate the outlier points around a candidate object. The modified seeded region growing method is able to robustly identify the same potential object points and recognise the same outlier points regardless of the initial selected seeds. It is also able to quickly identify a different number of neighbours for each data point based on its cloud density. At c_i location of a candidate cuboid in each section l, the points in $\mathbf{TS}_l^{(c_i)}$ may represent a traffic sign object or a false object. Similarly, the points in $\mathbf{LP}_l^{(c_i)}$ may represent a light pole object or a false object. Regardless of the object types, there might be some outlier points around the objects. To eliminate these outliers, we randomly select a seed point and grow a uniform connected region from this seed. A point that has not been assigned to any other region is added to the seeded region if and only if the point is in the neighbour of the region and its distance to the region is less than a predefined threshold. Otherwise, the point will be a new seed for another region.

To do so, for each $\mathbf{TS}_l^{(c_i)}$ or $\mathbf{LP}_l^{(c_i)}$, we calculate its *xz* projection. Any point in the *xz* plane is selected as the seed and its six nearest neighbours, which can fit a local plane, are then selected. A neighbour is added to the region containing the seed if its Euclidean distance to the centroid of the region is <0.5 m and it has not been assigned to any region. The algorithm continues to find the six nearest neighbours of each of the added points and repeats the same process to grow the region. When no neighbour can be added to any existing region, the algorithm finds a point that has not been assigned to any region as the seed and starts repeating the same growing process. We save all the points identified in the growing regions in a region list $\mathbf{RL}_l^{(c_i)}$. The algorithmic overview of the proposed modified region growing algorithm for LiDAR data points is presented in Algorithm 4 (see Fig. 6).

2.3.2 False object removal: We propose to use the shape information in the *xz*-plane to remove the points corresponding to false objects. Specifically, we find the largest connected component $\mathbf{ObjRegion}^{(c_i)}$ for points within each $\mathbf{RL}_l^{(c_i)}$. We then find the minimum bounding box $\text{BoundingBox}^{(c_i)}$ to cover $\mathbf{ObjRegion}^{(c_i)}$.

For the traffic sign class $\mathbf{TS}_l^{(c_i)}$, we find the convex hull $\text{ConvexHull}^{(c_i)}$ around the points in the corresponding $\mathbf{ObjRegion}^{(c_i)}$. We then compute the ratio of the area of $\text{BoundingBox}^{(c_i)}$ to the area of $\text{ConvexHull}^{(c_i)}$ to decide whether the traffic sign class corresponds to the false objects. If the ratio is larger than a predefined threshold (e.g. 0.7), it indicates that the two areas are similar and therefore we consider $\mathbf{ObjRegion}^{(c_i)}$ as a traffic sign. Otherwise, we consider $\mathbf{ObjRegion}^{(c_i)}$ as the false object and remove it from the final results.

For the light pole class $\mathbf{LP}_l^{(c_i)}$, we calculate the ratio of the length to the width of $\text{BoundingBox}^{(c_i)}$. If this ratio is larger than a predefined threshold (e.g. 5), it indicates that $\mathbf{ObjRegion}^{(c_i)}$ is a tall rectangle and therefore we consider $\mathbf{ObjRegion}^{(c_i)}$ as a light pole.

Input: The candidate set $\check{\mathbf{X}}_l^{(c_i)} \subseteq \mathbb{R}^4$ at c_i location of the cuboid in section l
Output: Region list of each candidate set in section l, $\mathbf{RL}_l^{(c_i)}$

1.Initialize the region list $\mathbf{RL}_l^{(c_i)}$ as empty.
2.Project the coordinate information (e.g., x, y, z) of the points in $\check{\mathbf{X}}_l^{(c_i)}$ onto the xz plane as $\mathbf{ProjectedRL}_l^{(c_i)}$
3.While $\mathbf{ProjectedRL}_l^{(c_i)}$ is not empty
(a)Initialize the queue $SeedSet$ as empty.
(b)Remove a point P_r in $\mathbf{ProjectedRL}_l^{(c_i)}$ and add P_r to $SeedSet$.
(c)While $SeedSet$ is not empty
(i)Remove the first element SP from $SeedSet$ and add SP to $\mathbf{RL}_l^{(c_i)}$.
(ii)Find 6 nearest neighbors b_j of SP $(1 \leq j \leq 6)$.
(iii)For each b_j $(1 \leq j \leq 6)$
•If $b_j \notin SeedSet$ and the distance from b_j to the centroid of the region is ≤ 0.5 meters
○Add b_j to $SeedSet$ and remove b_j from $\mathbf{ProjectedRL}_l^{(c_i)}$.
Endif
Endfor
Endwhile
Endwhile

Fig. 6 *Algorithm 4: modified seeded region growing algorithm*

Input: Region list $\mathbf{RL}_l^{(c_i)}$ at each candidate location c_i in each section l, a predefined threshold Th, and another predefined threshold S_R
Output: Traffic sign points $\mathbf{FinalTS}_{All}$ and light pole points $\mathbf{FinalLP}_{All}$.

1.For each $\mathbf{RL}_l^{(c_i)}$
(a)Find its largest connected component $\mathbf{ObjRegion}^{(c_i)}$.
(b)Find the minimum bounding box of $\mathbf{ObjRegion}^{(c_i)}$ as $BoundingBox^{(c_i)}$.
Endfor
2.For each $\mathbf{RL}_l^{(c_i)}$ in traffic sign class
(a)Compute the convex hull of $\mathbf{ObjRegion}^{(c_i)}$ as $ConvexHull^{(c_i)}$.
(b)If $\frac{area(BoundingBox)^{(c_i)}}{area(ConvexHull^{(c_i)})} \geq Th$
•Add $\mathbf{ObjRegion}^{(c_i)}$ to $\mathbf{FinalTS}_{All}$
EndIf
Endfor
3.For each $\mathbf{RL}_l^{(c_i)}$ in light pole class
(a)Compute the ratio of the length to the width of $BoundingBox^{(c_i)}$ as $R_C^{(c_i)}$.
(b)If $R_C^{(c_i)} \geq S_R$
•Add $\mathbf{ObjRegion}^{(c_i)}$ to $\mathbf{FinalLP}_{All}$.
EndIf
EndFor

Fig. 7 *Algorithm 5: false object removal*

Otherwise, we consider $\mathbf{ObjRegion}^{(c_i)}$ as the false object and remove it from the final results.

The algorithmic view of the proposed false object removal is provided in Algorithm 5 (see Fig. 7).

3 Experimental results

We evaluate the performance of the proposed traffic sign and light pole detection method by conducting various experiments on eight MLS point clouds. These LiDAR point clouds are collected and provided by the Utah Department of Transportation (UDOT). The data for this study were collected by a vehicle with the following equipment: Velodyne LiDAR sensor, laser road imaging system, laser rut and crack measurement system, road surface profiler, and position orientation system. Specifically, the data were collected on eight different one-mile sections of the I-15 highway, which may contain traffic signs, light poles, trees, bridges, billboards, buildings, and so on. We denote the name of each dataset by a three-digit number followed by a hyphen and another three-digit number. The first three-digit number shows the starting milepost of the I-15 highway and the second three-digit number shows the ending milepost. The names of the dataset in our experiments are

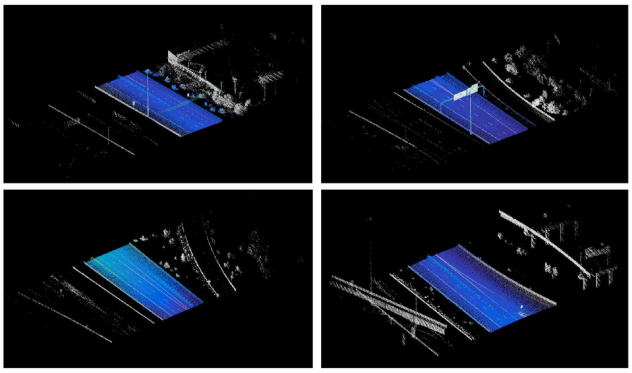

Fig. 8 *Illustration of the road extraction results for four representative segments 3, 5, 19, and 22 (in the raster scan order) of the dataset 304-305. Each section is a portion of the dataset along the trajectory of the vehicles*

304-305, *305-306*, *258-259*, *259-260*, *261-262*, *263-264*, *247-248*, and *262-263*, respectively. We manually select the ground truth of the road points. The ground truth of the traffic signs and light poles for each dataset is provided by UDOT experts. In this section, we present original MLS point clouds for a segment of the datasets *304-305* and *305-306* and demonstrate the road extraction results for four representative sections and the traffic sign and light pole extraction results for eight representative sections by employing the proposed method and its variant method without involving the post-processing step. Furthermore, we summarise both qualitative and quantitative results for all the eight datasets and compare the proposed method with several state-of-the-art methods.

3.1 Results on the first dataset 304-305

The first dataset *304-305* corresponds to highway I-15 milepost 304 to 305, which is located in the south of Salt Lake City of Utah. The direction of the road is from south to north. The area contains high density of traffic signs, light poles, trees, bridges, buildings, and billboards. The size of this MLS dataset is 157 MB. This dataset contains more than 4.7 million points.

As discussed in Section 2, we divide the dataset into $N = 24$ sections and separately process each section and concatenate the results of the sections to obtain the desired output. We empirically decide the optimal number of sections based on the road detection overlap score when utilising a different number of sections. Adopting the similar metric introduced in [28, 29], the road detection overlap score is computed as follows:

$$\text{Overlap} = \frac{|\mathbf{Result} \cap \mathbf{GT}|}{|\mathbf{Result} \cup \mathbf{GT}|} \tag{7}$$

where $|\cdot|$ returns the number of elements in an array, $\cap$ is the intersect operation, $\cup$ is the union operation, ***Result*** is the extracted interest points (e.g. road points), and **GT** is the manually labelled ground truth of interest points (e.g. road points).

We run the proposed road point extraction method using a different number of sections (i.e. $N = 2, 4, 8, 12, 16, 20, 24, 28, 32$) on the dataset *304-305* and calculate their corresponding road detection overlap score. The minimum road detection overlap score is 86.12% when $N = 2$ and the maximum road detection overlap score is 87.12% when $N = 24$. It is clear from these experiments that the number of sections does not significantly influence the road detection overlap score. However, we can achieve the most accurate road point extraction when setting N to be 24 for this dataset. Qualitatively, we present the road extraction results in blue on four sections of dataset *304-305* in Fig. 8 for $N = 24$. The results show the robustness of the proposed method in extracting the road points. However, several points such as some of the vegetation part in the top-left plot of Fig. 8 and road curbs in the bottom-left plot of Fig. 8 are incorrectly labelled as the road points (i.e. false positive points) due to their similarity in terms of altitude, intensity, and orientation.

Table 1 True positives and false positives of the proposed traffic sign and light pole extraction method and its variant method on the dataset *304-305*

Ground truth		Variant method			Proposed method		
Traffic sign	Light poles	Traffic signs	Light poles	False objects	Traffic signs	Light poles	False objects
23	8	22	8	25	21	8	9

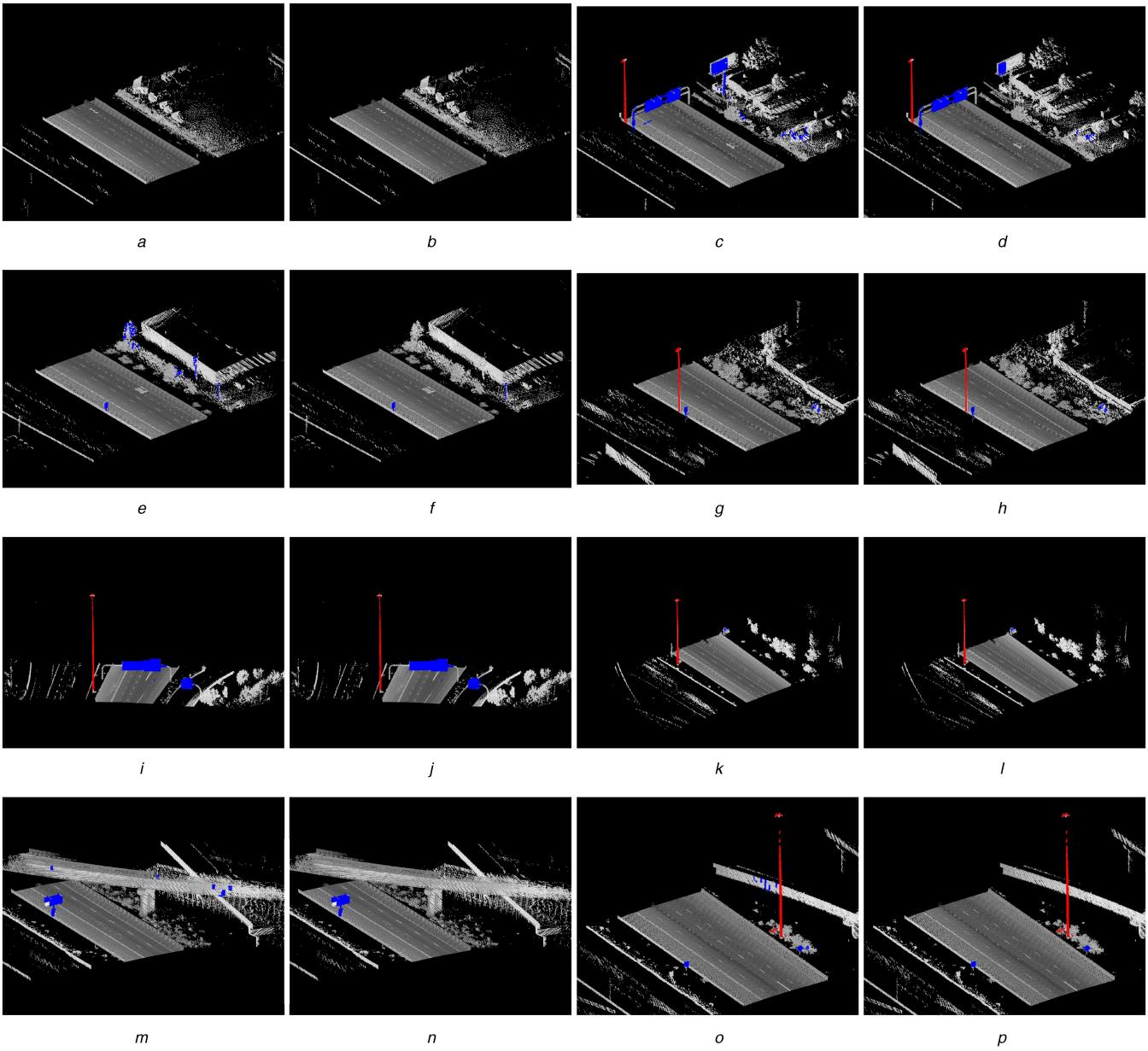

Fig. 9 *Traffic sign and light pole extraction results respectively shown in blue and red obtained by the variant method and the proposed method for eight representative sections of the dataset 304-305: Section 2 shown in (a) and (b), Section 3 shown in (c) and (d), Section 5 shown in (e) and (f), Section 8 shown in (g) and (h), Section 13 shown in (i) and (j), Section 15 shown in (k) and (l), Section 19 shown in (m) and (n), Section 22 shown in (o) and (p)*

We compare the extracted traffic signs and light poles with the ground truth of their counterparts to quantitatively evaluate the performance of the proposed method and its variant method without involving the post-processing step. There are 23 traffic signs and 8 light poles in this section of the highway. Table 1 lists the true positives (i.e. the number of correctly extracted traffic signs and light poles) of the proposed method and its variant method on the dataset *304-305*. It also lists the false positives (i.e. the number of incorrectly extracted traffic signs and light poles) of the two compared methods. The table clearly shows that the proposed method successfully extracts 21 (e.g. 91.30%) traffic signs and eight (e.g. 100%) light poles and its variant method successfully extracts 22 (e.g. 95.65%) traffic signs and eight (e.g. 100%) light poles. The proposed method extracts nine false objects (i.e. nine objects are incorrectly extracted as true objects) while its variant method extracts 25 false objects. It is clear that the two-step post-processing method significantly reduces the false positive rate. However, it also removes one correctly identified traffic sign from the desired output. Since it is not desirable to have high false positives and the post-processing step does keep a majority of the correctly extracted traffic signs and light poles, we decide to incorporate the post-processing step in the final proposed method.

We choose eight representative sections of the dataset *304-305* to qualitatively illustrate the traffic sign and light pole extraction results of the proposed method (i.e. with employing post-processing) and its variant method (i.e. without employing post-processing). These eight sections contain a varied number of traffic signs, light poles, and other objects such as billboard, poles, bridges, and trees. Fig. 9 presents the extracted traffic signs in blue and the extracted light poles in red on top of the original MLS

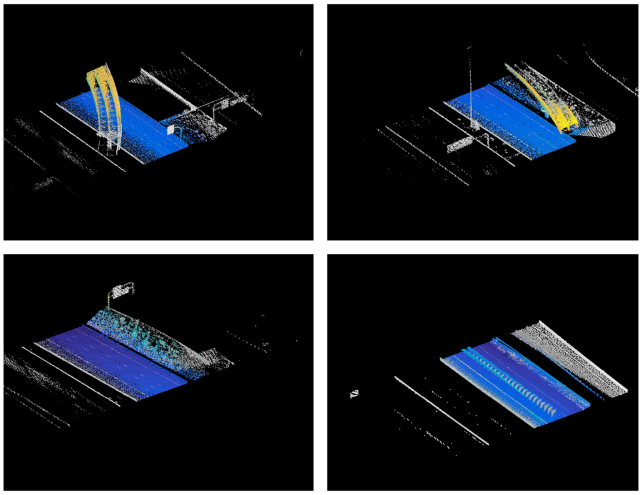

Fig. 10 *Illustration of the road extraction results for four representative segments 6, 7, 8, and 21 (in the raster scan order) of the dataset 305-306*

Table 2 True positives and false positives of the proposed traffic sign and light pole extraction method and its variant method on the dataset *305-306*

Ground truth		Variant method			Proposed method		
Traffic sign	Light poles	Traffic signs	Light poles	False objects	Traffic signs	Light poles	False objects
18	9	17	8	7	16	8	3

point clouds for the eight sections of the dataset *304-305*. Figs. 9*a* and *b* demonstrate the extraction results of the variant method and the proposed method for section 2, respectively. It clearly shows that both methods do not detect any non-target objects as target objects when there is no traffic signs and light poles along the road. Figs. 9*c* and *d*, respectively, demonstrate the extraction results of the variant method and the proposed method for section 3. It clearly shows that the variant method correctly detects one traffic sign and one light pole. However, it also mistakenly detects the billboard and some parts of trees at the right border of the road as traffic signs. The proposed method effectively removes some parts of trees from the extraction results. Figs. 9*e* and *f* present the extraction results of the variant method and the proposed method for section 5, respectively. It clearly shows that the variant method correctly detects the traffic sign located at the left border of the road and incorrectly detects some vegetation parts and two poles at the right side of the road as traffic signs. The proposed method effectively removes most of the tree parts and a pole at the right border of the road from the traffic sign extracting results. Similarly, the variant method accurately detects one traffic sign and one light pole in section 8 as shown in Fig. 9*g* and two traffic signs and one light pole in section 13 as shown in Fig. 9*i*. The proposed method achieves the same extraction results for sections 8 and 13 as shown in Figs. 9*h* and *j*, respectively. The variant method extracts one light pole in section 15 as shown in Fig. 9*k*, one traffic sign in section 19 as shown in Fig. 9*m*, and two traffic signs and one light pole in section 22 as shown in Fig. 9*o*. However, it also detects few false objects. For example, Fig. 9*k* shows that a part of the tree that is located at the right top corner of the road is detected as a traffic sign by mistake; Fig. 9*m* shows that some parts of the bridge are mistakenly detected as a traffic sign; and Fig. 9*o* shows that some part of the tree at the right border of the road and some parts of the bridge are misclassified as traffic signs. Figs. 9*l*, *n*, and *p* demonstrate the extraction results of the proposed method for the same sections 15, 19, and 22, respectively. We observe that the post-processing step effectively removes the parts of the bridge in section 19, and some of vegetation parts and bridge parts in section 22. However, some false objects (e.g. mostly part of bridges and billboards) are still present in the extraction results due to their similarity to the traffic signs.

3.2 Results on the second dataset 305-306

The second dataset *305-306* corresponds to highway I-15 milepost 305 to 306, which is located in the south of Salt Lake City of Utah. The direction of the road is from south to north. The area contains high density of traffic signs, light poles, bridges, buildings, and billboards. The size of this MLS dataset is 145 MB. This dataset contains more than 4.3 million points.

We run the proposed road point extraction method using a different number of sections (i.e. $N = 2, 4, 8, 12, 16, 20, 24, 28, 32$) on the second dataset and calculate their corresponding road detection overlap score. The minimum road detection overlap score is 86.12% when $N = 2$ and the maximum road detection overlap score is 86.27% when $N = 24$. Our extensive experimental results on the other datasets also show that setting $N = 24$ usually achieves the most accurate road point extraction results. Qualitatively, we present the road extraction results in blue on four sections of the dataset *305-306* in Fig. 10. The results show the robustness of the proposed method in extracting the road points. However, several points, such as a small part of the vegetation at the right border of the road shown at the top row of Fig. 10 and a part of vegetation shown at the bottom-left plot of Fig. 10, are incorrectly labelled as the road points.

We compare the extracted traffic signs and light poles with the ground truth of their counterparts to quantitatively evaluate the performance of the proposed method and its variant method. There are 18 traffic signs and 9 light poles in this section of the highway. Table 2 lists both the true positives and the false positives of the proposed method and its variant method on the dataset *305-306*. It clearly shows that the proposed method successfully extracts 16 (e.g. 88.89%) traffic signs and eight (e.g. 88.89%) light poles and its variant method successfully extracts 17 (e.g. 94.44%) traffic signs and eight (e.g. 88.89%) light poles. The proposed method extracts three false objects while its variant method extracts seven false objects. It is clear that the post-processing step reduces the number of false positives and keeps the majority of the true positives in the final extraction results. The traffic sign and light pole extraction results on datasets *304-305* and *305-306* demonstrate that the proposed method works well by correctly extracting almost all the target objects and obtaining fewer false positives.

Similar to the experimental results for the first dataset, we also choose eight representative sections of the dataset *305-306* to qualitatively illustrate the traffic sign and light pole extraction results of the proposed method (i.e. with employing post-processing) and its variant method (i.e. without employing post-processing). Fig. 11 presents the extracted traffic signs in blue and the extracted light poles in red on top of the original MLS point

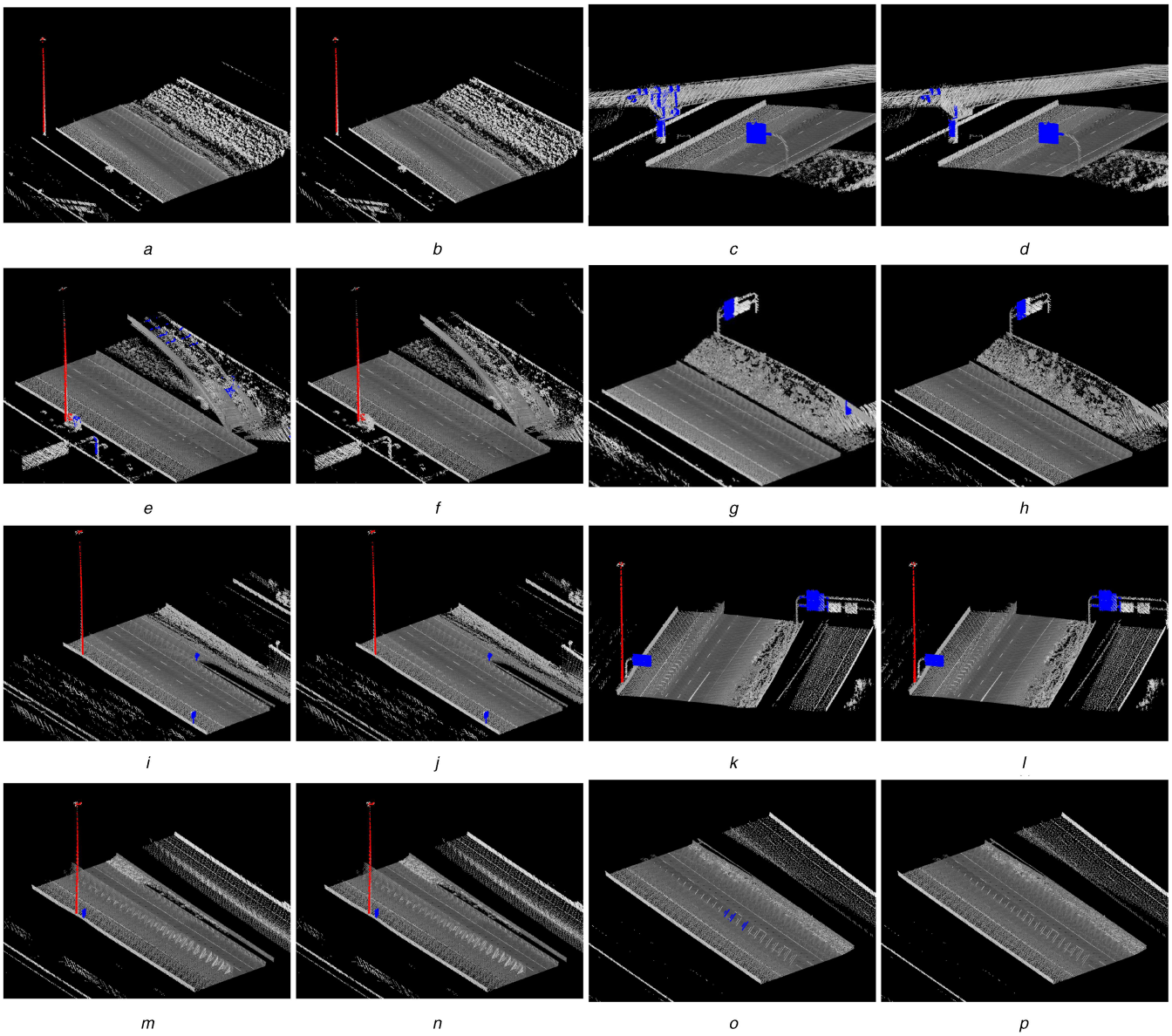

Fig. 11 *Traffic sign and light pole extraction results respectively shown in blue and red obtained by the variant method and the proposed method for eight representative sections of the dataset 305-306: Section 3 shown in (a) and (b), Section 6 shown in (c) and (d), Section 7 shown in (e) and (f), Section 8 shown in (g) and (h), Section 12 shown in (i) and (j), Section 15 shown in (k) and (l), Section 19 shown in (m) and (n), Section 21 shown in (o) and (p)*

clouds for the eight sections of *305-306*. These plots clearly demonstrate that the variant method accurately extracts the light pole in section 3 as shown in Fig. 11*a*, the traffic sign in section 6 as shown in Fig. 11*c*, and the light pole in section 7 as shown in Fig. 11*e*, one traffic sign at the right border in section 8 as shown in Fig. 11*g*, two traffic signs and one light pole in section 12 as shown in Fig. 11*i* and section 15 as shown in Fig. 11*k*, and one traffic sign and one light pole that are located at the left border of the highway in section 19 as shown in Fig. 11*m*. However, it misidentifies a part of the bridge shown in Figs. 11*c*, *e*, and *g*, and a part of a vehicle on the road shown in Fig. 11*o* as traffic sign. It also misses a part of the traffic sign in Figs. 11*g* and *k* since the missed traffic sign is not located along the main road of the highway. The proposed method achieves the same extraction results for sections 3, 12, 15, and 19 as it is shown in Figs. 11*b*, *j*, *l*, and *n*, respectively. We observe that the post-processing step effectively removes some parts of the bridge in Fig. 11*c*, all the bridge parts in Fig. 11*e*, a small part of the bridge at the bottom right border of the road in Fig. 11*g*, and a vehicle that is passing on the highway in Fig. 11*o*. However, some false objects (e.g. mostly part of the trees) are still present in the extraction results due to their similarity to the traffic signs.

3.3 Results on the other six datasets

To further evaluate the proposed method, we also conduct experiments on the remaining six datasets. Each of these six datasets contains the data along the one milepost of the highway I-15. We empirically choose N to be 24 to divide each dataset into 24 segments since this choice achieves the best road point extraction overlap score for all the six datasets.

Fig. 12 presents the final traffic sign and light pole extraction results after employing the proposed method on the six datasets, namely, *258-259*, *259-260*, *261-262*, *262-263*, *263-264*, and *247-248*. The traffic sign extraction results are plotted in blue and the light pole extraction results are plotted in red on top of the original MLS point clouds. Fig. 12*a* shows that the proposed method correctly extracts all the 16 traffic signs with different sizes that are located at the borders of the road in the dataset *258-259*. It detects two out of three light poles and mistakenly detects a part of the bridge at the end of the road and a pole-like object at the beginning of the road as a traffic sign. Fig. 12*b* shows that the proposed method correctly detects 19 out of 20 traffic signs in the dataset *259-260*, which are located at the borders of the road. It mistakenly detects the bridge that is located at the end of this section as a traffic sign. Fig. 12*c* shows that the proposed method detects 18 out of 20 traffic signs in the dataset *261-262*. However, it also detects three false objects as traffic signs. Two of these false

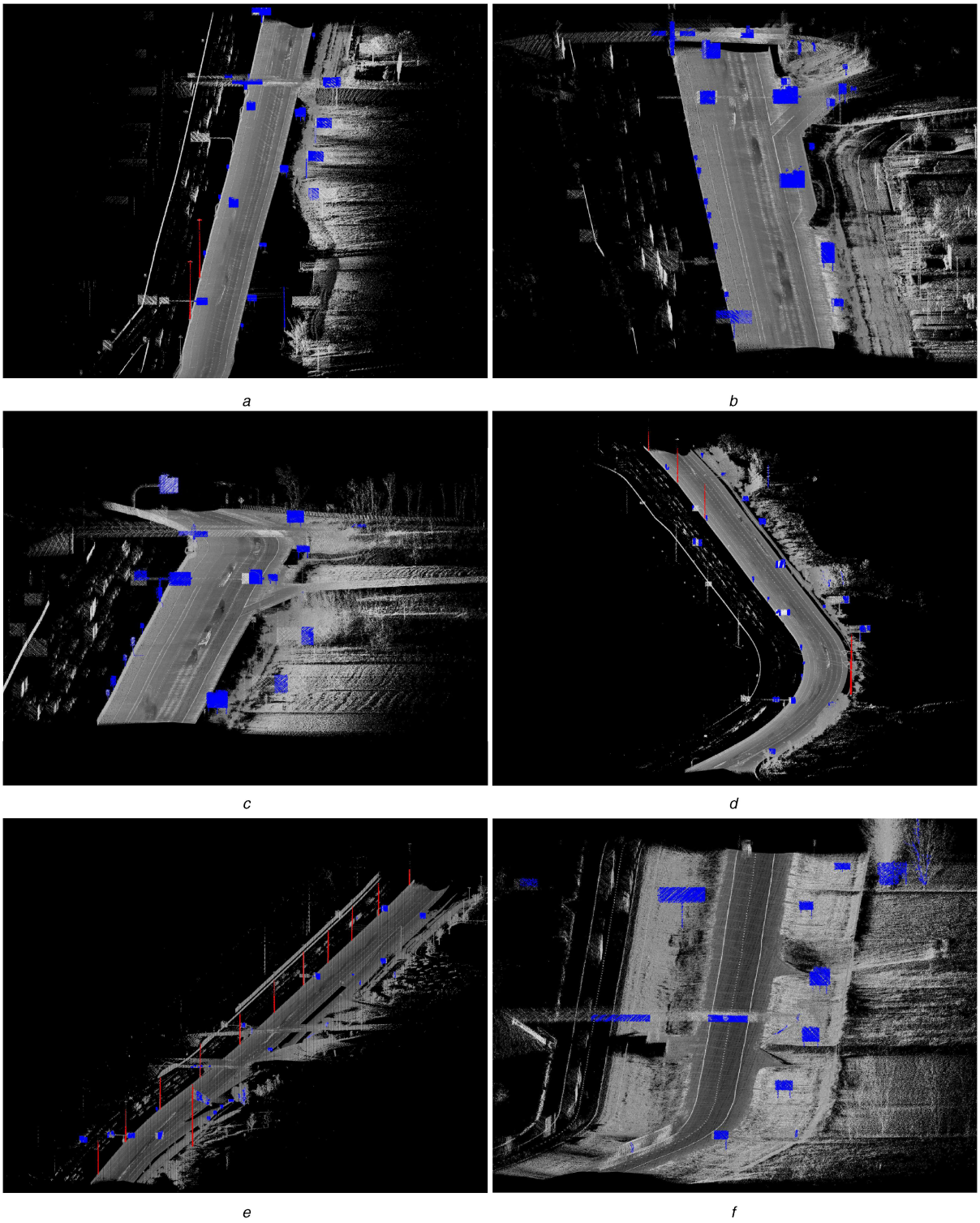

Fig. 12 *Illustration of traffic sign extraction results in blue and light pole extraction results in red after applying the proposed method on six datasets* ***(a)*** *258-259*; ***(b)*** *259-260*; ***(c)*** *261-262*; ***(d)*** *262-263*; ***(e)*** *263-264*, ***(f)*** *247-248*

objects are the billboards that are located near the border of the road and the other false object is a part of a bridge. Fig. 12*d* shows that the proposed method successfully detects all the 22 traffic signs and four out of five light poles in the dataset *262-263* and wrongly detects three false objects (e.g. a billboard, an off-road board, and a part of wires on top of the road) as traffic signs. Fig. 12*e* shows that the proposed method is capable of detecting 18 out of 19 traffic signs and 11 out of 12 light poles in the dataset *263-264*. It also detects six false objects. One of these false objects is a flag pole that exhibits the same characteristics as a light pole. Three of them are parts of trees and two are parts of bridges since they show the same characteristics as the traffic signs. Fig. 12*f* shows that the proposed method detects all seven traffic signs in the dataset *247-248* and detects two parts of a bridge, a billboard, and an off-road board by mistake.

3.4 Overall quantitative results on eight datasets

We compare the extracted traffic signs and light poles with the ground truth of their counterparts to quantitatively evaluate the performance of the proposed method and its variant method. Table 3 lists the ground truth and both the true positives and the false positives of the proposed method and its variant method on all eight datasets. It clearly shows that the proposed method successfully extracts 137 (e.g. 94.48%) traffic signs and 33 (e.g. 89.19%) light poles and its variant method successfully extracts

139 (e.g. 95.86%) traffic signs and 33 (e.g. 89.19%) light poles. The proposed method incorrectly extracts 29 false objects while its variant method incorrectly extracts 70 false objects. It is clear that the proposed method detects most of the traffic signs and light poles prosperously. Moreover, it significantly reduces the number of false objects due to the use of the post-processing step.

To further evaluate the proposed method, we provide the confusion matrix along with six evaluation measures including the number of missed objects, the number of false objects, recall (detection rate), precision, *quality* [30], and F_1-*measure* [31] in Table 4. This confusion matrix shows that the proposed method correctly classifies the traffic signs and light poles in their corresponding classes. However, some false objects are predicted as traffic signs or light poles mistakenly. The proposed method also misses few objects (i.e. eight traffic signs and four light poles). The average recall, precision, *quality* and F_1-*measure* of traffic sign and light pole extraction results are 91.84, 87.85, 81.31, and 89.68%, respectively.

In addition, we evaluate the running time of the proposed method for the aforementioned eight datasets. The proposed method is written in Matlab 2017(b) and the program is run on Intel®-Core™ i7-3370 (3.4 GHz) system with 16 GB RAM. Table 5 lists the number of MLS points and the approximate running time in seconds for the proposed method on each of the eight datasets. The average running time of the proposed method for 4.423×10^6 MLS points is 191 s.

Finally, we evaluate the effectiveness of the adaptive threshold Θ (the threshold calculated in Algorithm 3 (see Fig. 4)), which is used to classify each object candidate as a traffic sign or a light pole. To evaluate the sensitivity of the proposed method to the choices of threshold values, we manually select different values and compute the precision and recall for traffic sign and light pole classes. Specifically, we calculate the precision and recall values of the traffic sign and light pole classes by using several fixed thresholds 5, 10, 15, and 20 and the adaptive threshold Θ, respectively. Table 6 demonstrates the effectiveness of the adaptive threshold Θ. It clearly shows that the adaptive threshold consistently achieves the best detection results when compared to the fixed thresholds. On average, the proposed method improves the method with the best threshold by 3.79% in precision and 4.62% in recall. The chosen adaptive threshold automatically decides the optimal threshold, which works well on each dataset.

3.5 Comparison with previous studies

Comparing the performance of the proposed method with the performance of previous studies is challenging due to the differences in datasets and the variation of the defined tasks. Datasets can be different in terms of point clouds' quality, density, distribution, and the areas that the data is collected from. For instance, one dataset can be a high-density MLS point clouds collected from city areas while the other can be an airborne laser scanning data collected from a highway. The detection and

Table 3 True positives and false positives of the proposed traffic sign and light pole extraction method and its variant method on all the eight datasets

Dataset	Ground truth		Variant method			Proposed method		
	Traffic sign	Light pole	Traffic sign	Light pole	False object	Traffic sign	Light pole	False object
258-259	16	3	16	2	4	16	2	2
259-260	20	0	19	0	3	19	0	1
261-262	20	0	18	0	6	18	0	3
262-263	22	5	22	4	6	22	4	3
263-264	19	12	18	11	12	18	11	5
247-248	7	0	7	0	7	7	0	3
304-305	23	8	22	8	25	21	8	9
305-306	18	9	17	8	7	16	8	3
Total	145	37	139	33	70	137	33	29

Table 4 Confusion matrix along with six evaluation measures for each of the two classes in all eight datasets

		Predicted	
		Traffic sign	Light pole
Actual	Traffic sign	137	0
	Light pole	0	33
missed objects		8	4
false objects		26	3
recall		94.48%	89.19%
precision		84.04%	91.67%
quality		80.11%	82.50%
F_1-*measure*		88.95%	90.41%

Table 5 Running time of the proposed method on each of eight datasets

Dataset	Number of points	Time in seconds
259-260	4.568×10^6	185
261-262	4.364×10^6	192
262-263	4.160×10^6	183
263-264	4.453×10^6	201
247-248	4.726×10^6	184
304-305	4.730×10^6	220
306-306	4.393×10^6	192
Average	4.423×10^6	191

Table 6 Evaluation of the proposed method using different classification thresholds

Threshold	Traffic sign		Light pole		Average	
	Precision, %	Recall, %	Precision, %	Recall, %	Precision, %	Recall, %
5	78.57	68.27	42.18	72.97	60.37	70.62
10	82.06	82.06	51.66	83.78	66.86	82.92
15	81.76	89.65	83.78	83.78	82.77	86.71
20	81.06	94.48	88.23	81.08	84.64	87.78
Θ	84.04	94.48	91.67	89.19	87.85	91.84

Table 7 Summary of precision and recall rates for four traffic sign detection methods

Compared methods	Precision rate, %	Recall rate, %
[32]	58.00	65.00
[8]	**95.74**	60.81
[12]	93.94	65.96
proposed	84.04	**94.48**

The highest precision and recall are shown in bold.

Table 8 Summary of precision and recall rates for five light pole detection methods

Compared methods	Precision rate, %	Recall rate, %
[32]	45.00	62.00
[33]	72.00	82.00
[8]	89.58	86.87
[12]	**94.29**	80.49
proposed	91.67	**89.19**

The highest precision and recall are shown in bold.

classification tasks are also not identical in different works. For instance, the objective in one work can be detection of lamp posts and pole-like objects while the objective in another work can be detection of traffic signs and pedestrians. Despite these challenges, we aim to demonstrate the performance of the proposed traffic sign and light pole detection method in comparison with the previous studies. To do so, we only include the object detection studies of street areas that use MLS point clouds and report the recall and precision of their targeted classes.

When comparing traffic sign extraction results, the proposed method is comparable with the three methods proposed in [8, 12, 32]. In [32], the authors only report the recall rate of 65% and the precision rate of 58% for 60 traffic signs. The method proposed in [8] achieves the recall of 60.81% and the precision of 95.74%. Specifically, out of the 74 traffic signs in their dataset, it correctly detects 45 traffic signs, mistakenly classifies 24 traffic signs to other classes, misses the remaining 5 traffic signs, and mis-detects 2 objects as traffic signs. Lehtomäki *et al.* [12] report the recall of 65.96% and the precision of 93.94%. Specifically, out of the 94 traffic signs in their dataset, their method correctly detects 62 traffic signs, erroneously classifies 11 traffic signs to other classes, misses the remaining 21 traffic signs, and incorrectly detects 4 objects as traffic signs. Our proposed method achieves 94.48% recall and 84.04% precision. Specifically, out of the 145 traffic signs, it correctly detects 137 of them and misses only 8 of them. It does not classify the traffic signs to other classes. However, it mis-detects 26 objects as traffic signs. The proposed method achieves a higher precision rate than the method proposed in [32] and the highest recall rate among the other three methods. Overall, it achieves the best traffic sign detection performance in terms of recall and precision. Table 7 summarises the precision and recall rates for four aforementioned traffic sign detection methods.

Different studies focus on different types of light poles. For example, the tasks in [8, 12, 32, 33] are to detect the standard light poles or lamp posts in city areas while the task in our study is to detect the light poles along the highway areas. The standard light poles in city areas usually have lower elevation than the ones in highway areas. Due to this difference, we compare the proposed method with the previous studies by comparing recall and precision rates. Golovinskiy *et al.* [32] only report the recall rate of 62% and the precision rate of 45% for 51 light poles. The method proposed in [33] achieves the recall rate of 82% and the precision rate of 72% for 73 light poles. The method proposed in [8] achieves 86.87% recall and 89.58% precision. Specifically, it correctly detects 86 of 99 pole objects, mistakenly classifies 5 of them to other classes, misses the remaining 8 pole objects, and mis-detects 10 objects as pole objects. The method proposed in [12] achieves the recall rate of 80.49% and the precision rate of 94.29% for lamp posts. Specifically, it detects 33 of 41 lamp posts correctly, classifies one of them incorrectly to another class, misses 7 of them, and detects 2 false objects as lamp posts. The proposed method achieves the recall rate of 89.19% and correctly detects 33 out of 37 light poles. It mis-detects three false objects as light poles and does not classify the light poles to the other classes. It achieves the second highest precision rate of 91.67% and the highest recall rate among the other four previous methods. Overall, it achieves the best light pole detection performance among the four compared previous methods. Table 8 summarises the precision and recall rates for five aforementioned light pole detection methods.

In summary, the proposed method is able to simultaneously detect both traffic signs and light poles in the MLS data. It achieves the best overall recall and precision rates in detecting both traffic signs and light poles when comparing with three traffic sign detection methods and four light pole detection methods. Furthermore, it achieves the highest recall rate in detecting both traffic signs and light poles. It achieves the second best precision rate in detecting light poles and a comparable precision rate in detecting traffic signs due to misclassification of several false objects as the target objects.

4 Conclusions

In this paper, we propose a fast and reliable traffic sign and light pole detection method, which can be applied to the MLS data to quickly identify various traffic signs and light poles. A set of experiments have been carried out on the eight datasets that are captured by UDOT along the I-15 highway. The extensive experimental results demonstrate that the proposed method is able to successfully detect 137 (e.g. 94.48%) traffic signs and 33 (e.g. 89.19%) light poles in the eight datasets. In other words, the proposed method is robust in detecting almost all the traffic signs and light poles with few number of false positives.

Our contributions are: (i) employing the surface reconstruction algorithm to extract the orientation of the points as one of the characteristic features; (ii) applying the unsupervised *k*-means clustering algorithm to automatically extract road points; (iii) designing a sliding cuboid to search for the high elevated objects above or beside the roads as groups of candidate points; (iv) employing the RANSAC algorithm to select the robust candidate points that represent planes along the vehicle trajectory; (v) proposing a modified seeded region growing algorithm to remove the outlier points around the objects; (vi) introducing a shaped-based false object rejection algorithm to remove the false positive objects.

In the future, we will improve the performance of the proposed method by designing an efficient image segmentation algorithm to identify the traffic signs and light poles with less computational time and more accuracy. We may consider incorporating de-noising techniques in the pre-processing step to further improve the segmentation results. We will utilise other information such as

colour and texture to remove more false objects to improve the precision rate.

5 Acknowledgment

The study was sponsored by Utah Department of Transportation (UDOT). The views expressed are those of the authors and do not reflect the official policy or position of the project's sponsor.

6 References

[1] Hernández, J., Marcotegui, B.: 'Point cloud segmentation towards urban ground modeling'. Urban Remote Sensing Event, Shanghai, China, 2009, pp. 1–5

[2] Zhu, X., Zhao, H., Liu, Y., *et al.*: 'Segmentation and classification of range image from an intelligent vehicle in urban environment'. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS), Taipei, Taiwan, 2010, pp. 1457–1462

[3] Douillard, B., Underwood, J., Kuntz, N., *et al.*: 'On the segmentation of 3D LiDAR point clouds'. IEEE Int. Conf. Robotics and Automation (ICRA), Shanghai, China, 2011, pp. 2798–2805

[4] Lin, C.H., Chen, J.Y., Su, P.L., *et al.*: 'Eigen-feature analysis of weighted covariance matrices for LiDAR point cloud classification', *ISPRS J. Photogramm. Remote Sens.*, 2014, **94**, pp. 70–79

[5] Soheilian, B., Paparoditis, N., Vallet, B.: 'Detection and 3d reconstruction of traffic signs from multiple view color images', *ISPRS J. Photogramm. Remote Sens.*, 2013, **77**, pp. 1–20

[6] Adam, A., Ioannidis, C.: 'Automatic road sign detecion and classification based on support vector machines and hog descriptors', *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.*, 2014, **2**, (5), p. 1

[7] Khalid, S., Muhammad, N., Sharif, M.: 'Automatic measurement of the traffic sign with digital segmentation and recognition', *IET Intell. Transp. Syst.*, 2018

[8] Pu, S., Rutzinger, M., Vosselman, G., *et al.*: 'Recognizing basic structures from mobile laser scanning data for road inventory studies', *ISPRS J. Photogramm. Remote Sens.*, 2011, **66**, (6), pp. S28–S39

[9] Yokoyama, H., Date, H., Kanai, S., *et al.*: 'Detection and classification of pole-like objects from mobile laser scanning data of urban environments', *Int. J. CAD/CAM*, 2013, **13**, (2), pp. 31–40

[10] Yu, Y., Li, J., Guan, H., *et al.*: 'Automated extraction of urban road facilities using mobile laser scanning data', *IEEE Trans. Intell. Transp. Syst.*, 2015, **16**, (4), pp. 2167–2181

[11] Riveiro, B., Díaz Vilariño, L., Conde Carnero, B., *et al.*: 'Automatic segmentation and shape-based classification of retro-reflective traffic signs from mobile LiDAR data', *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, 2016, **9**, (1), pp. 295–303

[12] Lehtomäki, M., Jaakkola, A., Hyyppä, J., *et al.*: 'Object classification and recognition from mobile laser scanning point clouds in a road environment', *IEEE Trans. Geosci. Remote Sens.*, 2016, **54**, (2), pp. 1226–1239

[13] El Halawany, S.I., Lichti, D.D.: 'Detection of road Poles from mobile terrestrial laser scanner point cloud'. 2011 Int. Workshop on Multi-Platform/ Multi-Sensor Remote Sensing and Mapping (M2RSM), Xiamen, China, 2011, pp. 1–6

[14] Zhang, K., Zuo, W., Chen, Y., *et al.*: 'Beyond a Gaussian denoiser: residual learning of deep cnn for image denoising', *IEEE Trans. Image Process.*, 2017, **26**, (7), pp. 3142–3155

[15] Muhammad, N., Bibi, N., Jahangir, A., *et al.*: 'Image denoising with norm weighted fusion estimators', *Pattern Anal. Appl.*, 2017, **21**, (4), pp. 1013–1022

[16] Hoppe, H., DeRose, T., Duchamp, T., *et al.*: '*Surface reconstruction from unorganized points*', vol. **26** (ACM, USA, 1992)

[17] Arthur, D., Vassilvitskii, S.: 'k-means++: the advantages of careful seeding'. Proc. Eighteenth Annual ACM-SIAM Symp. Discrete Algorithms, New Orleans, LA, USA, 2007, pp. 1027–1035

[18] Torr, P.H., Zisserman, A.: 'Mlesac: a new robust estimator with application to estimating image geometry', *Comput. Vis. Image Underst.*, 2000, **78**, (1), pp. 138–156

[19] Krizhevsky, A., Sutskever, I., Hinton, G.E.: 'Imagenet classification with deep convolutional neural networks'. Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 2012, pp. 1097–1105

[20] He, K., Zhang, X., Ren, S., *et al.*: 'Deep residual learning for image recognition'. Proc. IEEE Conf. Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 2016, pp. 770–778

[21] Qi, C.R., Su, H., Mo, K., *et al.*: 'Pointnet: deep learning on point sets for 3d classification and segmentation'. Proc. Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017

[22] Mahjoub, H.N., Tahmasbi Sarvestani, A., Kazemi, H., *et al.*: 'A learning-based framework for two-dimensional vehicle maneuver prediction over v2v networks'. IEEE 15th Int. Dependable, Autonomic and Secure Computing, 15th Int. Conf. Pervasive Intelligence & Computing, 3rd Int. Conf. Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech), 2017, Orlando, FL, USA, 2017, pp. 156–163

[23] Akram, T., Laurent, B., Naqvi, S.R., *et al.*: 'A deep heterogeneous feature fusion approach for automatic land-use classification', *Inf. Sci.*, 2018, **467**, pp. 199–218

[24] Simonyan, K., Zisserman, A.: 'Very deep convolutional networks for largescale image recognition', Proc. Int. Conf. on Learning Representation, San Diego, CA, USA, 2015

[25] Russakovsky, O., Deng, J., Su, H., *et al.*: 'Imagenet large scale visual recognition challenge', *Int. J. Comput. Vision (IJCV)*, 2015, **115**, (3), pp. 211–252

[26] Everingham, M., Eslami, S.A., Van Gool, L., *et al.*: 'The pascal visual object classes challenge: a retrospective', *Int. J. Comput. Vis.*, 2015, **111**, (1), pp. 98–136

[27] Yi, L., Kim, V.G., Ceylan, D., *et al.*: 'A scalable active framework for region annotation in 3D shape collections', *ACM Trans. Graph.*, 2016, **35**, (6), p. 210

[28] Wu, Y., Lim, J., Yang, M.H.: 'Online object tracking: a benchmark'. Proc. IEEE Conf. Computer Vision and Pattern Recognition, Portland, OR, USA, 2013, pp. 2411–2418

[29] Wu, Y., Lim, J., Yang, M.H.: 'Object tracking benchmark', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2015, **37**, (9), pp. 1834–1848

[30] Rutzinger, M., Rottensteiner, F., Pfeifer, N.: 'A comparison of evaluation techniques for building extraction from airborne laser scanning', *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, 2009, **2**, (1), pp. 11–20

[31] Guan, H., Li, J., Yu, Y., *et al.*: 'Using mobile laser scanning data for automated extraction of road markings', *ISPRS J. Photogramm. Remote Sens.*, 2014, **87**, pp. 93–107

[32] Golovinskiy, A., Kim, V.G., Funkhouser, T.: 'Shape-based recognition of 3d point clouds in urban environments'. IEEE 12th Int. Conf. Computer Vision, Kyoto, Japan, 2009, pp. 2154–2161

[33] Velizhev, A., Shapovalov, R., Schindler, K.: 'Implicit shape models for object detection in 3D point clouds'. Int. Society of Photogrammetry and Remote Sensing Congress, Melbourne, Australia, 2012, vol. 2