ISPRS Journal of Photogrammetry and Remote Sensing 128 (2017) 111–129

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

SigVox – A 3D feature matching algorithm for automatic street object recognition in mobile laser scanning point clouds

Jinhu Wang [a,b,*], Roderik Lindenbergh [a], Massimo Menenti [a]

[a] Dept. of Geoscience and Remote Sensing, Delft University of Technology, Building 23, Stevinweg 1, Post Box 5048, 2628CN Delft, The Netherlands
[b] Key Laboratory of Quantitative Remote Sensing Information Technology, Academy of Opto-Electronics, Chinese Academy of Sciences, No. 9 Dengzhuang South Road, HaiDian District, 100094 Beijing, China

ARTICLE INFO

Article history:
Received 31 July 2016
Received in revised form 19 January 2017
Accepted 21 March 2017
Available online 3 April 2017

Keywords:
Mobile laser scanning
Point cloud
Octree
Multiple scale feature
Object recognition
3D feature descriptor

ABSTRACT

Urban road environments contain a variety of objects including different types of lamp poles and traffic signs. Its monitoring is traditionally conducted by visual inspection, which is time consuming and expensive. Mobile laser scanning (MLS) systems sample the road environment efficiently by acquiring large and accurate point clouds. This work proposes a methodology for urban road object recognition from MLS point clouds. The proposed method uses, for the first time, shape descriptors of complete objects to match repetitive objects in large point clouds. To do so, a novel 3D multi-scale shape descriptor is introduced, that is embedded in a workflow that efficiently and automatically identifies different types of lamp poles and traffic signs. The workflow starts by tiling the raw point clouds along the scanning trajectory and by identifying non-ground points. After voxelization of the non-ground points, connected voxels are clustered to form candidate objects. For automatic recognition of lamp poles and street signs, a 3D significant eigenvector based shape descriptor using voxels (SigVox) is introduced. The 3D SigVox descriptor is constructed by first subdividing the points with an octree into several levels. Next, significant eigenvectors of the points in each voxel are determined by principal component analysis (PCA) and mapped onto the appropriate triangle of a sphere approximating icosahedron. This step is repeated for different scales. By determining the similarity of 3D SigVox descriptors between candidate point clusters and training objects, street furniture is automatically identified. The feasibility and quality of the proposed method is verified on two point clouds obtained in opposite direction of a stretch of road of 4 km. 6 types of lamp pole and 4 types of road sign were selected as objects of interest. Ground truth validation showed that the overall accuracy of the ~170 automatically recognized objects is approximately 95%. The results demonstrate that the proposed method is able to recognize street furniture in a practical scenario. Remaining difficult cases are touching objects, like a lamp pole close to a tree.

1. Introduction

Urban roads are of crucial importance in modern society by reducing the distance between people and services, and produce economic and social benefits (Vanier, 2006). The condition of the urban road furniture, i.e. street lamps, traffic lights, traffic signs, bus station signs and billboards, needs to be inspected and documented regularly to avoid potential risks caused by wear, vandalism or accidents (Halfawy, 2008). Furthermore, high precision urban maps are extensively demanded in various fields, such as smart cities (Nebiker et al., 2010; Batty et al., 2012), autonomous driving (Li et al., 2004; Schreiber et al., 2013) and intelligent transportation systems (Bishop, 2000; Agamennoni et al., 2011; Ivan et al., 2015), etc. Efficient and frequent updating of the urban road inventory is essential to ensure the overall technical and social function of a city. Currently, safety inspections on roadside furniture are conducted by manual in situ examination or semi-automatic interpretation of collected imagery and video data (Pu et al., 2011). These methods are valid and practical in identifying defect roadside furniture. However, the methods are labor intensive and expensive. Therefore, the methods are not optimal for striking a balance between maintaining safety and reducing expenses.

In the last decades new techniques based on photogrammetry and remote sensing have been developed for obtaining accurate

* Corresponding author.
E-mail address: jinhu.wang@tudelft.nl (J. Wang).

http://dx.doi.org/10.1016/j.isprsjprs.2017.03.012

3D urban measurements (Haala and Brenner, 1999; Ellum and El-Sheimy, 2002; Frueh and Zakhor, 2003; Over et al., 2010; Mc Elhinney et al., 2010; Puente et al., 2013). Among the developed techniques, Mobile Laser Scanning (MLS) systems, which combine Light Detection And Ranging (LiDAR), Global Navigation Satellite Systems (GNSS) and an Inertial Navigation System (INS), are able to obtain dense and highly accurate point measurements (Vosselman and Maas, 2010). A MLS system continuously samples the road environment and the geometry of objects is captured in form of point clouds with versatile information, i.e. precise coordinates, intensity and color. In recent years, the collected point cloud data have been used in various applications, such as 3D tree detection and modeling (Rutzinger et al., 2010; Zhong et al., 2013; Wu et al., 2013; Lindenbergh et al., 2015), road surface extraction (Jaakkola et al., 2008; Mc Elhinney et al., 2010; Pu et al., 2011; Wang et al., 2013; Guan et al., 2014), curbstone identification (Zhou and Vosselman, 2012; Yang et al., 2012, 2013; Kumar et al., 2014), road corridor objects classification (Pu and Zhan, 2009; Puttonen et al., 2011), change detection (Qin and Gruen, 2014) and mountain road monitoring (Wang et al., 2014; Díaz-Vilarino et al., 2016). Particularly, the obtained high density point clouds enable the detection and identification of objects along road corridors. Pole-like objects, such as tree trunks, lamp poles and traffic light poles can be identified and extracted (Brenner, 2009; Golovinskiy et al., 2009; Lehtomäki et al., 2010; Pu et al., 2011; Yang et al., 2012, 2015; Cabo et al., 2014). However, still lacking are methods for recognition, identification and grouping specific types of roadside object from MLS point clouds.

In this work, an automatic urban roadside object recognition method is presented for MLS point cloud data. The proposed method starts with raw point cloud data obtained by a MLS system. First, the point cloud is tiled along the road direction following the trajectory of the MLS system. Next, the point cloud tiles are divided in ground and non-ground points. The non-ground points are organized in an octree data structure and connected voxels are clustered. Consecutively, a newly proposed 3D SigVox shape descriptor of the objects of interest, such as different types of street lamps and traffic signs, is constructed. Finally, objects in the clustered point clouds are recognized by SigVox descriptor enabled template matching.

The contribution of this work to the state of the art is as follows: (i) It introduces a novel 3D multi-scale shape descriptor, that is easy to compute and powerful for shape detection; (ii) It gives a workflow to use this shape descriptor to identify different types of lamp poles and traffic signs; and (iii) In doing so, it shows how to efficiently handle large MLS point clouds, e.g. by using a suitable tiling strategy. What we consider as a notable innovation is that this method, for the first time, use shape descriptors of complete objects to match repetitive objects in large point clouds.

The remainder of the article is organized as follows. Section 2 presents related work in object recognition, followed by a detailed description of the proposed method in Section 3. In Section 4 the proposed method is demonstrated and validated on a MLS point cloud sampling 4 km of road environment. Finally, conclusions and recommendations are given in Section 5.

2. Related work

MLS systems efficiently sample the surface of objects along a road and record the measurements as dense and accurate point clouds (Puente et al., 2013; Barber et al., 2008; Haala et al., 2008; Cahalane et al., 2010). The acquired measurements, which typically consist of 3D coordinates, intensity and color information, enable the recognition of roadside objects. A variety of methods has been presented on this topic. The available methods for object recognition can roughly be classified into three categories: (i) model fitting based (Pu et al., 2011; Rutzinger et al., 2010; Lehtomäki et al., 2010; Brenner, 2009; Cabo et al., 2014; Xiao et al., 2016); (ii) semantic based (Fan et al., 2014; Yang et al., 2015; Babahajiani et al., 2015); and (iii) shape based (Golovinskiy et al., 2009; El-Halawany and Lichti, 2011; Velizhev et al., 2012; Bremer et al., 2013; Yang and Dong, 2013; Li and Oude Elberink, 2013; Rodríguez-Cuenca et al., 2015). This section first reviews the related work corresponding to the aforementioned methods on object recognition in Sections 2.1,2.2,2.3 respectively. Finally, a comparison of some related methods is given in a table.

2.1. Model fitting methods

A model fitting based object recognition method in general starts with segmenting and clustering the point cloud, followed by fitting the point segments to known geometric models, such as cylinders and planes. Brenner developed an algorithm for pole extraction from MLS scanned point clouds. The method first assumes that the basic characteristic of a pole is that it is upright. There is a kernel region where laser scanned points are present and an outside region where no points are present (Brenner, 2009). Point segments are analyzed in cylindrical stacks and when a certain minimum number of stacks is detected, the segment is considered as a pole. The final step is to estimate the exact position of the pole.

Lehtomäki et al. presented an algorithm to detect pole-like objects in a road environment using MLS point clouds. The algorithm first segmented scan lines and then remaining point groups were clustered. Consecutively adjacent clusters that are closely connected or overlapping in horizontal profiles were merged. Based on these merged point clusters, cylinder fitting was performed to detect poles along the road direction. The method was able to find 77% of the poles if compared to a manual data analysis with a correctness of 81% (Lehtomäki et al., 2010).

Pu et al. presented a method to recognize basic structures from MLS point clouds for road inventory. The method first roughly classified tiled raw point clouds into ground and non-ground objects. Then geometric attributes, i.e. size, position, orientation, color and material were characterized and organized per segment. Consecutively objects with planar features were approximated by planar models, such as rectangles, circles and triangles. Pole-like objects were sliced vertically and for each slice a 2D enclosing rectangle was derived. Next the differences of those rectangles, i.e. position and size differences, between neighboring slices were checked, and if they were within a defined threshold then similar slices were accumulated. Finally, if the number exceeded a maximum length, a point segment was considered a pole-like object (Pu et al., 2011). This method was capable of recognizing building walls and pole-like objects such as lamp poles and tree trunks. The final results showed that 86% and 64% of poles and trees respectively were correctly recognized.

Cabo et al. introduced an algorithm that automatically detects pole-like street furniture objects from MLS point clouds. Rather than directly considering each point as the aforementioned methods did, this algorithm first simplified the point cloud into voxels. Then each 2D vertical layer of the constructed 3D regular voxels was analyzed and potential elements were selected by an isolation criterion. The isolation criterion was evaluated based on fitting two 2D rings of different radii. If a candidate voxel cluster is enclosed between the inner and outer ring, then it is considered as a potential pole object. The algorithm was tested on point clouds of four test sites and was able to recognize all the target pole-like objects except of severely occluded ones (Cabo et al., 2014).

Xiao et al. did not consider pole-like objects but introduced a method to detect street-side vehicles with a deformable vehicle

model using MLS point clouds (Xiao et al., 2016). This method first classified raw point clouds into ground, buildings and street objects. Then geometric features were extracted from obtained street objects. Next, these features were fit to an explicit model. The vehicle recognition accuracy could reach up to 95%.

2.2. Semantic methods

Semantic methods for object recognition usually define a set of rules based on prior knowledge of the objects. Then based on these rules, objects are extracted and recognized. Fan et al. introduced an algorithm for identifying man-made objects along urban road corridors from MLS point clouds (Fan et al., 2014). The method assumes that, firstly, man-made objects are geometric regular whereas vegetation has diversity in shape; secondly, different urban man-made objects are characterized by the point extension and the height above the ground level. With the above rules, the method divides a MLS point cloud into three layers with respect to vertical height. In each layer, seeds of man-made object are indicated by a line filter in the foot prints of off-ground objects. Further classification is performed on those seeds by checking in which layers the seed points of an object are found. Finally, points belonging to respective objects are retrieved based on the classified seed points. The capability on extracting man-made objects was found to have a detection rate of 83%.

Teo et al. proposed a similar method as Cabo et al. in (Cabo et al., 2014) to detect pole-like object from MLS point clouds (Teo and Chiu, 2015). After removal of building facade points, the point cloud was resampled by voxels which were used to obtain a coarse segmentation. Next, fine-segmentation is conducted based on point to point distances which enables the separation of overlapping objects. Based on a series of predefined rules, pole-like objects were detected in a hierarchical way. The method was tested on two point clouds and the results showed that the correctness of the pole-like detection are 97.8% and 96.3% respectively for those test data sets. However, the method cannot classify different type of pole-like objects.

Babahajiani et al. presented a method to recognize objects in 3D point clouds of urban street environments (Babahajiani et al., 2015). The method starts with automatically extracting ground points. Building facades are detected using binary range images. Then the remaining points are voxelized and transformed to super voxels. Consecutively, boosted decision trees are employed to train and classify the extracted local 3D features of the voxel cells. The output of the classification is labeled with semantic classes. This method is evaluated on a challenging fixed-position TLS point cloud and a MLS point cloud. The global accuracy and per-class accuracy were about 94% and 87% respectively.

Yang et al. proposed an automatic algorithm for hierarchical extraction of urban objects from MLS point clouds (Yang et al., 2015). The method segments MLS points into ground and non-ground points. Based on the non-ground points, multi-scale supervoxels are generated. For each supervoxel, its geometric nature is determined by PCA. Then the multi-scale supervoxels are segmented with regard to their geometric type. In addition, the saliency of the segments is also calculated. Furthermore, seven semantic rules are defined corresponding to seven types of object, i.e. building, utility poles, traffic signs, trees, streetlamps, enclosures and cars. The method was validated on two MLS point clouds and the results demonstrate that the object extraction and classification accuracy of the proposed method was better than 91%.

2.3. Shape based methods

Shape based methods consider the explicit or implicit shapes of point clusters or segments. Then shape features are calculated to classify and identify objects from MLS point clouds. Golovinskiy et al. presented a shape-based recognition method for analyzing 3D MLS point clouds of urban environments (Golovinskiy et al., 2009). The method first determines the location of each potential object, then those objects are segmented from the original point cloud. Features are extracted from these object segments. Classification of those segments is performed based on the extracted features. The evaluation demonstrated that this method is able to recognize 65% of the objects.

El-Halawany et al. proposed a pipeline for roadside pole detection from MLS point clouds (El-Halawany and Lichti, 2011). The algorithm first calculates the eigenvalues of the covariance matrix of a local neighborhood using a KD tree. Then eigenvalue-based segmentation is conducted and linear objects are extracted by region growing. The final recognition results are evaluated by cylinder fitting and an eigen-radius relation. Velizhev et al. proposed an implicit shape model based automatic method for object localization and recognition in 3D outdoor scenes from MLS point clouds (Velizhev et al., 2012). The method consists of two steps. First a list of hypotheses on objects is determined by connected component extraction. Then objects are recognized using local descriptors and a voting-based localization method. The method was validated on a MLS point cloud and the recognition precision was 68% and 72% for cars and light poles respectively.

Bremer et al. presented a method based on eigenvalues and graphs to extract objects from MLS point clouds (Bremer et al., 2013). First the method calculates a 3×3 covariance matrix for each point from a local neighborhood and eigenvalues and eigenvectors are derived. Then those eigenvalues are characterized and classified. By connected component segmentation and clustering, ground and building facades are separated. Finally pole objects including trees are separated using a Dijkstra region growing approach.

Yu et al. proposed a street light pole extraction algorithm for MLS point clouds based on pairwise 3D shape context (Yu et al., 2015). The method first detects road curbstones based on a series of profiles perpendicular to the road direction. Curb-lines are extracted to divide a point cloud into road and non-road points. Ground points are further segmented from non-road points using a voxel based height filter. Next, non-ground points are clustered as separated object segments. Finally a point based 3D shape context, i.e. the fast point feature histogram (FPFH) proposed in (Rusu et al., 2009), was used to match the objects of interest. The method was tested on a MLS point cloud and street poles were robustly extracted with a completeness over 99% and a correctness of 97%.

Rodríguez-Cuenca et al. presented an automatic pole-like object detection and classification method from MLS and TLS point clouds based on an anomaly detection algorithm. The method first extracts ground points and then based on an anomaly detection algorithm, vertical objects are detected as point clusters. Then the detected vertical objects are classified as either man-made poles or trees. The testing results demonstrated that the detection rate was 96% and the classification rate was 95%.

Table 1 summarizes some key methods and compares their approach to the proposed SigVox method. So far, only (Yu et al., 2015) also matches different instances of the same type of furniture. Our proposed method has the same goal, but we propose the use a shape descriptor that operates at object scale.

3. Methodology

As illustrated in Fig. 1, the proposed algorithm for automatic urban roadside object recognition consists of four consecutive steps:

Table 1
A comparison of four existing methods with the proposed method.

Method	Shape descriptor	Identified objects	Used input	Voxel
Yang et al.	No	Buildings, traffic signs, trees, street lamps, cars, enclosures	(x, y, z, Intensity, r, g, b)	No
Yu et al.	Point based	Bus stations, light poles traffic poles	(x, y, z)	No
Cabo et al.	No	Poles	(x, y, z)	Yes
Teo et al.	No	Pole-like objects	(x, y, z)	Yes
SigVox	Object based	Different lamp poles and road signs	(x, y, z)	Yes

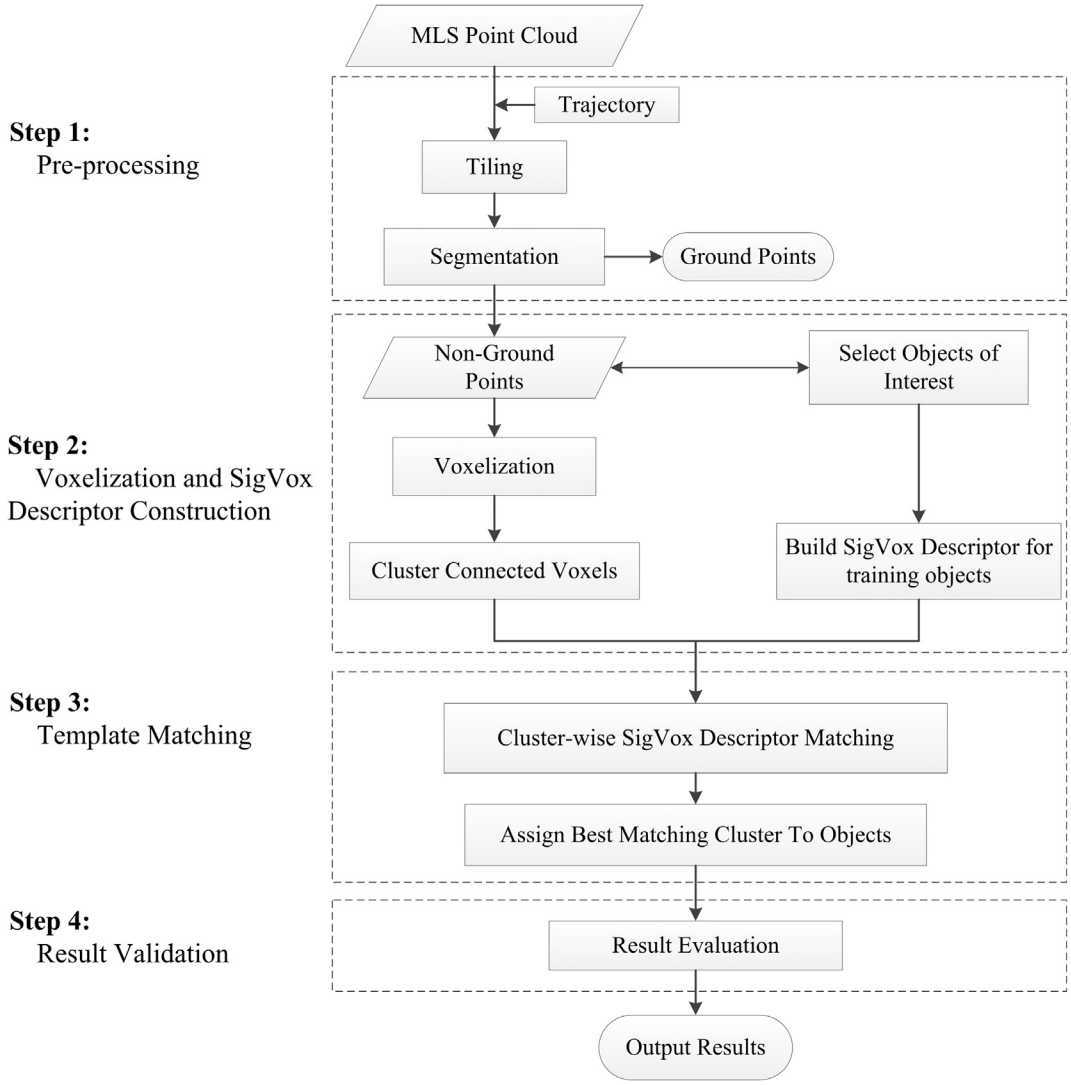

Fig. 1. Overall methodology of the proposed algorithm. The method starts with a MLS point cloud and results in a list of roadside objects.

1. Pre-processing. First the input raw point cloud is tiled with regard to the scanning trajectory. Next, the tiled point cloud is divided in non-ground and ground points.
2. Voxelization and building SigVox descriptors. Non-ground points are voxelized using an octree and connected voxels are clustered. Examples of objects of interest are manually selected for training. SigVox descriptors are constructed to form a template list.
3. Similarity Matching. Each of the clusters is automatically examined if it is a candidate for the selected objects of interests. If yes, then its SigVox descriptor is built and its similarity to the different training objects is computed. The cluster is then assigned to the best matched training object.
4. Validation. The recognition results are analyzed with regard to ground truth data and their accuracy is determined.

3.1. Pre-processing

The pre-processing in this work consists of two parts, i.e. tiling of the raw point cloud, and separation of ground and non-ground points, as indicated in Fig. 1. One scan of a MLS point cloud data set usually is too large to process on a normal desktop computer. Thus the raw point cloud is divided into tiles of suitable size. Furthermore, the focus of this work is on the non-ground objects rather than the ground points. Therefore, the tiles of the point cloud are further segmented into ground and non-ground points.

In this work, the scanning trajectory of the MLS system is used to partition the original MLS point cloud. Trajectory data is obtained by the Position and Orientation System (POS) of the MLS system, which consists of a series of 3D positions recorded at high frequency. The original MLS point cloud is tiled along the scanning trajectory.

Fig. 2 shows an example tiling of a raw point cloud acquired by a MLS system. The red line is a segment of the so-called Smoothed Best Estimation of Trajectory (SBET) of the MLS system and the purple arrow indicates the scanning direction. In this example, three tiles are generated and overlapping areas are indicated. During tiling, the 3D trajectory is projected on the horizontal plane. For each tile, the length along trajectory, i.e. the distance between *Starting point* and *Endpoint* along SBET in Fig. 2, and the width across trajectory are flexible. Fig. 3 is a magnification of the 2D polygon in Fig. 3. Points P_1, P_2 and P_3 are points of SBET and d is the width of the polygon in the across trajectory direction. The 2D boundary of each tile is obtained by accumulating polygons of the defined resolution k. In the first polygon $L_1R_1R_2L_2$, edge L_2R_2 is perpendicular to line segment P_1P_2. Consecutively, edge L_3R_3 of the second polygon $L_2R_2R_3L_3$ is obtained. All those concatenated 2D polygons form the boundary of *Tile 1*. Finally, all points that project within the boundary of the tile are output as belonging to that tile.

The next step of pre-processing is to identify the non-ground points in the tiled point clouds. In this study, the algorithm proposed by Pfeifer (Pfeifer, 2001) is used. The non-ground points will be forwarded to the next step. Meanwhile, points from objects of interest, such as street lamp poles and traffic signs, are selected and stored separately for training purposes.

3.2. Voxelization

The voxelization is performed on the non-ground points only. In this step, non-ground points are re-sampled to voxels organized by an octree data structure. Next connected voxels are clustered.

The octree data structure is extended to 3D from the 2D quadtree as introduced by Klinger (Allen, 1971). The octree data structure connects each branch of the tree node with a 3D Cartesian node, which is also defined as a voxel. This tree node can be subdivided recursively into 8 branches, i.e. octants. Fig. 4 demonstrates an octree based Cartesian spatial subdivision and its hierarchical data structure. This data structure enables efficient neighborhood searching by building up a series of loop up tables. In this work, the neighborhood searching strategy proposed by Payeur is implemented (Payeur, 2006).

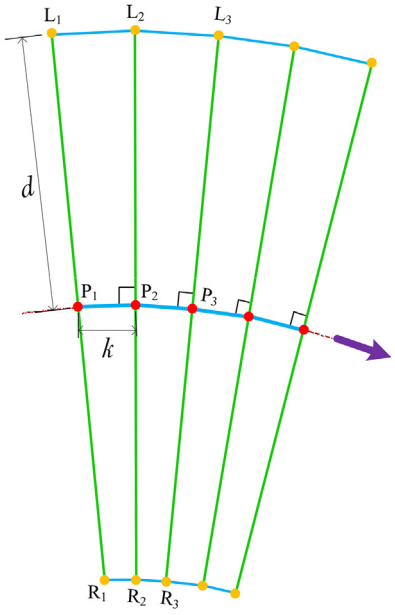

Fig. 3. Geometry of a tile polygon along the scanning trajectory. The P_i denote given trajectory points. The L_i and R_i are computed tile boundary points and are defined on the basis of a width parameter d.

3.3. Clustering and selecting candidate clusters

For the voxelization of the non-ground points, the bounding box of a tile of non-ground points is considered as the root node of the octree structure. The root node is subdivided recursively until the preset subdivision criteria are met. Consecutively, connected voxels are clustered based on a 3D seed filling algorithm proposed by Yu et al. (Yu et al., 2010). Then the points inside each voxel cluster are stored as point clusters. The subsequent point clusters are denoted by $C^i, i = 1, 2, \ldots, k$. Here k is the number of obtained point clusters. It is expected that many of these point clusters correspond to a roadside object like a street pole or traffic sign.

Next the 3D bounding box, i.e. $B(C^i)$, of each point cluster C^i, is obtained and compared with the 3D bounding boxes of the selected training objects T^j, $j = 1, 2, \ldots, h$. Here h is the number of training objects of interest. If the relative difference between the 3D bounding box of the j-th point cluster and the 3D bounding box of a training object is small, the point cluster is considered as a

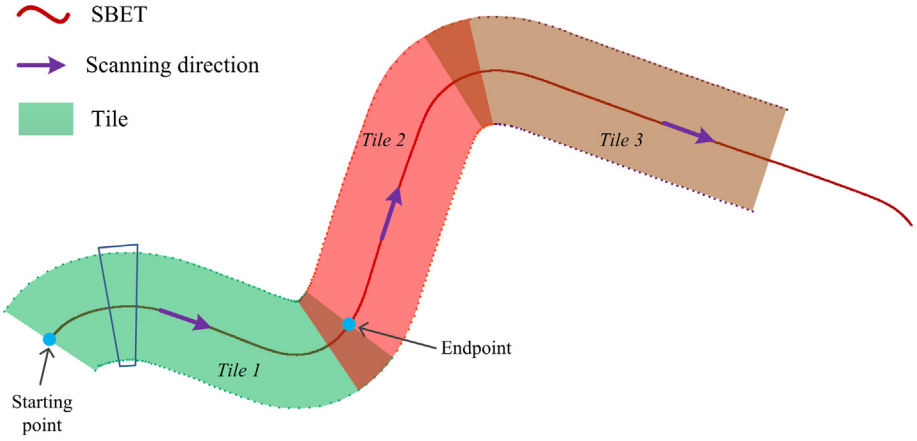

Fig. 2. Tiling of original MLS point cloud data along the road corridor direction. Different colors indicate different tiles. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

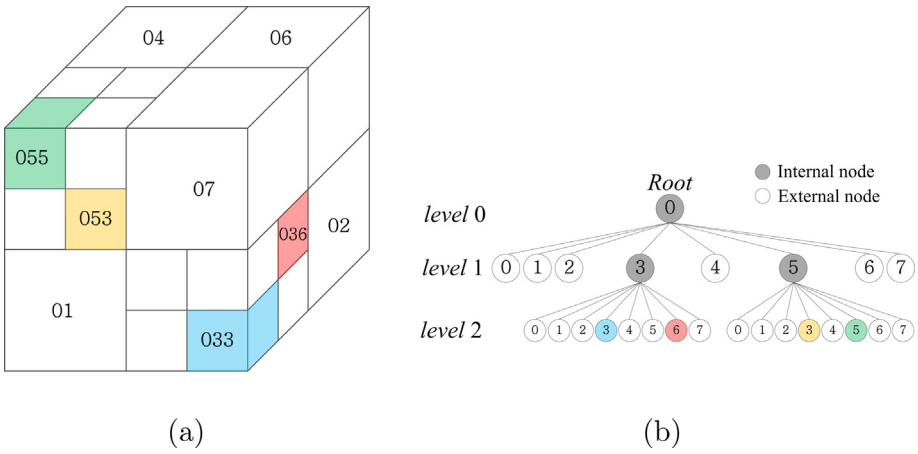

Fig. 4. Octree based space partition and its hierarchical data structure. (a) Octree based Cartesian space subdivision and voxel indexing. (b) Octree hierarchical data structure. The octant numbered 055 in the top left of (a) indicates where in the hierarchy as in (b) this voxel can be found.

candidate of the j-th object. The relative difference $D_{i,j}$ between two 3D bounding boxes is computed by Eq. (1).

$$D_{i,j} = \sqrt{\frac{\left\|B(C^i) - B(T^j)\right\|}{\left\|B(T^j)\right\|}} \quad (i = 1,2,\ldots,k; j = 1,2,\ldots,h)$$
$$= \sqrt{\frac{[B(C^i)_x - B(T^j)_x]^2 + [B(C^i)_y - B(T^j)_y]^2 + [B(C^i)_z - B(T^j)_z]^2}{[B(T^j)_x]^2 + [B(T^j)_y]^2 + [B(T^j)_z]^2}} \tag{1}$$

Here, $B(C^i)$ and $B(T^j)$ are the 3D bounding boxes of a candidate cluster and a training object respectively. Suppose there are k candidate clusters and h training clusters. B_x, B_y and B_z are the sizes of the bounding box in the three coordinate directions. When $D_{i,j}$ is smaller than a preset threshold, then a point cluster is regarded as a candidate of the j-th training object $C(T^j)$. Due to possible variations in the orientation of the roadside object, the orientation of bounding box will vary as well. Thus, the threshold used here for candidate object selection needs to be big enough to avoid omission.

3.4. SigVox descriptor construction

In this section, the approach for determining the dimensionality of a voxel is given. Consecutively, the concept of the 3D SigVox descriptor and the methodology to construct SigVox descriptors is presented.

3.4.1. Dimensionality analysis

Non-ground objects were clustered as voxel clusters in Section 3.2. Consecutively the dimensionality of each voxel in these clusters is determined by PCA. The dimension of a voxel is determined as follows: Suppose $p_i = (x_i \quad y_i \quad z_i)^T$ are the coordinates of a point p_i inside the voxel cell, then the center of gravity $\overline{p}$ of all the points p_i inside the voxel is determined by Formula 2.

$$\overline{p} = \frac{1}{n}\sum_{i=1}^{n} p_i \tag{2}$$

Here n is the number of points inside the voxel. The 3D structure tensor M of the points is defined by Formula 3.

$$M = \frac{1}{n} Q^T Q \tag{3}$$

Here $Q = (p_1 - \overline{p}, p_2 - \overline{p}, \ldots, p_n - \overline{p})^T$. M is a symmetric matrix and can be decomposed as $M = RIR^T$. Here R is a rotation matrix and I a diagonal positive definite matrix. The elements of I are the eigenvalues of matrix M. The three eigenvalues are positive, are denoted by λ_1, λ_2, λ_3, and are sorted such that $\lambda_1 > \lambda_2 > \lambda_3$. The corresponding eigenvectors are v_1, v_2, v_3 respectively.

In this work, voxel cells are categorized into three types: linear, planar and scatter. The three cases are defined as follows: (i) If for the eigenvalues of a voxel it holds that $\lambda_1 \gg \lambda_2$, then this voxel is defined as a linear, or 1D voxel cell. For a linear voxel cell, eigenvector v_1, which corresponds to eigenvalue λ_1, is the significant eigenvector and indicates the dominant direction of the points inside the voxel cell. (ii) If $\lambda_2 \gg \lambda_3$, then the voxel cell is defined as a planar, or 2D cell. In this case eigenvector v_3, the normal vector of the plane, is the significant eigenvector. (iii) If $\lambda_1 \approx \lambda_2 \approx \lambda_3$, then this cell is defined as a scatter cell, or 3D cell. A scatter cell does not have a dominant direction and will not be considered. Here $\gg$means *much larger* and is implemented by a preset threshold. In this work, the linearity is examined first and then planarity. The thresholds for linearity and planarity are denoted by T_l and T_p respectively. After this procedure, all voxel cells in a cluster have a geometric flag denoting its dimensionality and if applicable, a significant eigenvalue and eigenvector as its properties.

3.4.2. EGI descriptor

In this section, the approach to construct the proposed SigVox 3D descriptor for both training objects and candidate voxel clusters is demonstrated. The SigVox descriptor is inspired by the existing EGI descriptor, proposed by Horn Horn (1984). The EGI descriptor in this work is approximated by an icosahedron. Rather than computing local normals from a query point with a radius as in Horn (1984), the significant eigenvectors obtained in Section 3.4.1 are used to construct a 3D EGI descriptor (Wang et al., 2016). An approximated sphere, i.e. an icosahedron in this work, is used to assign significant eigenvectors to its surface and is also defined as the *Eigen-Sphere* of a voxel cluster.

The full sphere is approximated by an icosahedron. As illustrated in Fig. 5, the relative position of the icosahedron with regard to the Cartesian coordinate axes is given by its standard position in this work. In Fig. 5, point O is the origin of the coordinate system and the geometric center of the icosahedron. The X axis intersects one icosahedron edge at point P_m and the Y axis penetrates trough one triangle patch at point Q_m, while the Z axis passes through vertex u. This is an uniform icosahedron, as the distances of its 12 vertices to the origin O are equal to 1.

The next procedure is to assign the obtained significant eigenvectors of all voxels in a cluster to the triangles on the boundary of the icosahedron. In Fig. 6, triangle *uae* is one of the 20 boundary

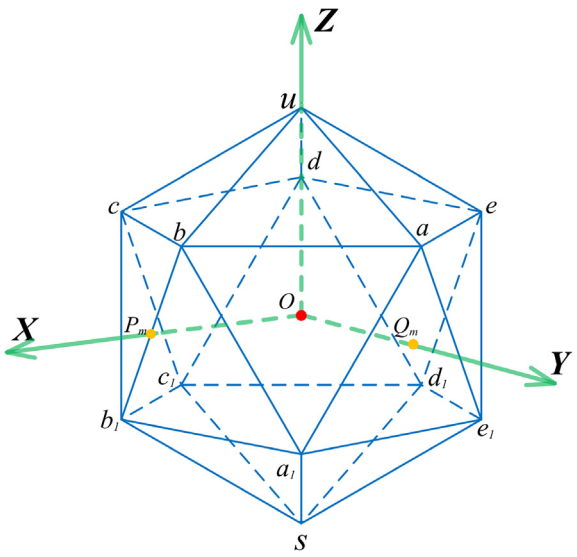

Fig. 5. A uniform icosahedron used to construct the EGI descriptor. Such icosahedron provides a coarse sphere approximation.

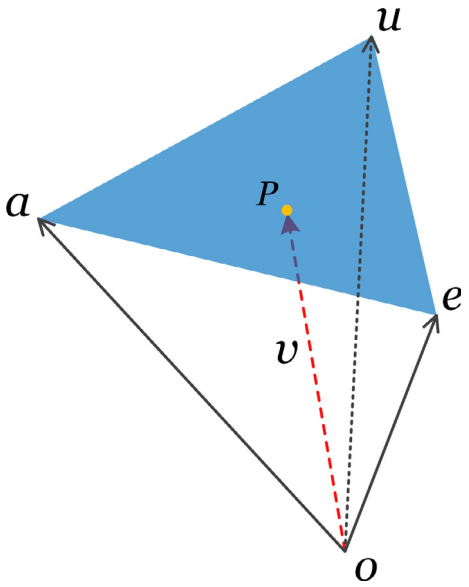

Fig. 6. Assign a significant eigenvector $\vec{v}$ to a triangle *aeu* on the boundary of the icosahedron.

triangles of the icosahedron in Fig. 5. Vector $\vec{v}$ intersects triangle *uae* at point *P*. Suppose the three vertices correspond to vectors $\vec{v_n}$, $\vec{v_a}$, $\vec{v_e}$ respectively, then the coefficients k_n, k_a, k_e in the linear combination in Eq. (4) must be all positive (Preparata and Shamos, 1985). This is used to assign significant eigenvectors to the correct triangle.

$$\vec{v} = k_n \vec{v_u} + k_a \vec{v_a} + k_e \vec{v_e} \tag{4}$$

Indeed, to determine the coefficients in Eq. (4), the function can be decomposed and rewritten to Eq. (5).

$$\begin{pmatrix} k_n \\ k_a \\ k_e \end{pmatrix} = \begin{pmatrix} v_u^{\rightarrow} & v_a^{\rightarrow} & v_e^{\rightarrow} \end{pmatrix}^{-1} \cdot \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix} \tag{5}$$

Here $\vec{v_n} = (x_n, y_n, z_n)^T$ are the coordinates of vertex v_n and v_x, v_y, v_z are the coordinates of vector $\vec{v}$.

The 3D EGI descriptor of the candidate cluster is constructed by assigning the significant eigenvectors of all voxels in the cluster to its *Eigen-Sphere* by determining for each eigenvector which triangle of the icosahedron it intersects using Eq. (5). This means that only the voxel eigenvector corresponding to the biggest eigenvalue is assigned for a linear voxel, while for a planar voxel, only its normal vector is assigned. Note that the eigenvectors obtained by PCA are not direction definite, e.g. vector $\vec{v}$ and its antipodal vector $-\vec{v}$ are both applicable. In this work, for the purpose of symmetrical concern, both a vector and its antipodal vector are assigned to the *Eigen-Sphere*.

For each significant eigenvector assigned to a particular triangle, a weight value W^k is stored, indicating the percentage of points it contains of the cluster it is from. That is, the weight W^k of a significant voxel contributing to the *i*-th triangle, is computed by Eq. (6).

$$W^k = \frac{N_k}{N} \tag{6}$$

Here N_k is the number of points in the considered voxel while N is the total number of points in the cluster.

The aim of the weights is to avoid ambiguity in dimensionality determination in Section 3.4.1. Division of the point cluster of an object using an octree will separate different parts of the object in a somewhat arbitrary way, depending on the particular orientation of the object. For example, a voxel that contains only one border of a plane will appear linear. Such voxels will lead to ambiguity in the dimensionality as a whole. The weight will take the number of points in the voxels into consideration. This weight is therefore designed to reduce the sketched ambiguity.

3.4.3. SigVox descriptor

In this work, a significant eigenvector based shape descriptor using multiple levels of voxel is proposed, which is denoted by SigVox. SigVox is constructed based on the recursive subdivision of each candidate point cluster using the octree. At each level of subdivision, the geometric dimensionality feature of each voxel cell is computed as described in Section 3.4.1 and their significant eigenvectors are obtained.

Fig. 7a shows a typical type of street lamp in this study. Fig. 7b is the corresponding point cloud of the pole and its original octree octant, which is also the root node of the octree.

Fig. 8 demonstrates the recursive subdivision of the street lamp pole by an octree at four levels and the corresponding *Eigen-Spheres*. In each sub-figure, the left figure denotes the subdivision of the point cluster while the right figure is the corresponding EGI descriptor represented by an *Eigen-Sphere*. In the sub-figures, linear, planar and scatter voxels are denoted in red, green and blue respectively. The triangles of the icosahedron at each subdivision level are colored according to the number of collected significant eigenvectors. The number of voxels at the four subdivision levels are 6, 10, 18 and 36 respectively. The number of significant voxels is 5, 8, 16 and 33 respectively. For example, in Fig. 8b the subdivision is at level 2 and there are 10 voxels. The two blue voxels are scatter voxels. Thus there are $10 - 2 = 8$ significant voxels and their eigenvectors are assigned to the three red and green triangles of the icosahedron.

For each candidate point cluster, their SigVox is defined as an ordered series of significant eigenvector based EGI at different levels of subdivision, as given by Formula 7.

$$SigVox = \{E_1, E_2, \ldots, E_n\} \tag{7}$$

Here, E_1, E_2, E_n are the corresponding EGI descriptors at level 1, 2 and *n*.

In this work, the similarity between each candidate point segment and its corresponding template segment will be determined

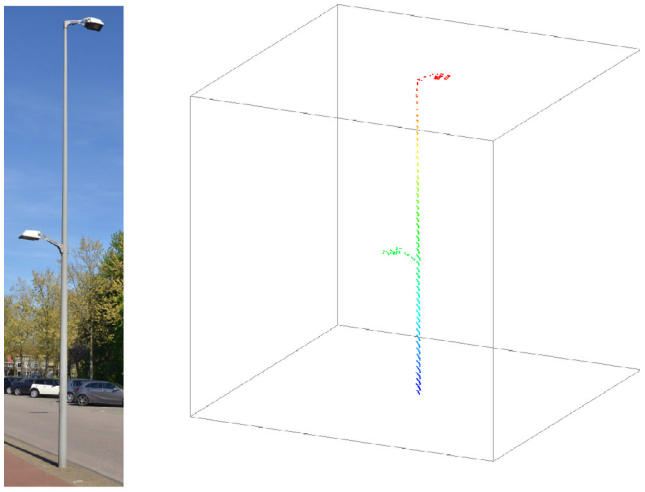

Fig. 7. A street lamp pole and its point cloud. (a) A typical street lamp. (b) Point cloud of the lamp pole and its original octree octant.

by comparing the distance between their SigVox descriptors for a preset number of levels.

3.5. Descriptor matching

This section will first introduce the distance between a pair of SigVox descriptors of two point clusters. The second part describes the transformation method that is used to evaluate the similarity between two EGI descriptors, i.e. the similarity at a fixed scale.

3.5.1. Distance between SigVox descriptors

The distance between the SigVox descriptors of candidate point clusters and template clusters is determined by accumulating the difference between the assigned vectors and their weights for each EGI triangle at each recursive subdivision level. An icosahedron has a total of 60 symmetry transformations (Conway et al., 2008). Since both the significant vector and their antipodal vector are assigned to the icosahedron surface, only 30 symmetries have to be considered in practice.

Suppose P_c and P_t denote a candidate point cluster and a template cluster respectively. Let E_c^l and E_s^t denote their EGI descriptor at level l respectively. Their similarity at level l is defined by Formula 8

$$\begin{aligned} S_l^{c,t} &= min\left\{\widehat{S}_l^{c,t}\right\}_j (j = 1, 2, \ldots, 30) \\ &= min\left\{\sum_{i=1}^{20} (N_{vec,c}^{i,l} * W_c^{i,l} - N_{vec,t}^{i,l} * W_t^{i,l})^2\right\}_j \end{aligned} \tag{8}$$

Here, $S_l^{c,t}$ denotes the similarity between point cluster P_c and template cluster P_t at level l. This final similarity is the best match among a total of 30 comparisons. That is, each $\widehat{S}_l^{c,t}$ gives the similarity for one among a total of 30 similarity transformations of the icosahedron. $N_{vec,c}^{i,l}$ and $N_{vec,t}^{i,l}$ are the number of eigenvectors that intersect the i-th triangle of the EGI from P_c and P_t at level l respectively. $W_c^{i,l}$ and $W_t^{i,l}$ are the weights of the i-th triangle from P_c and P_c at level l. Here j denotes the j-th symmetry of the icosahedron. The similarity is defined by the minimum distance of the similarity among all 30 icosahedron similarities.

The multiple scale distance between a pair consisting of a point cluster P_c and template cluster P_t is simply the sum of the similarities at all subdivision levels, as denoted by Formula 9.

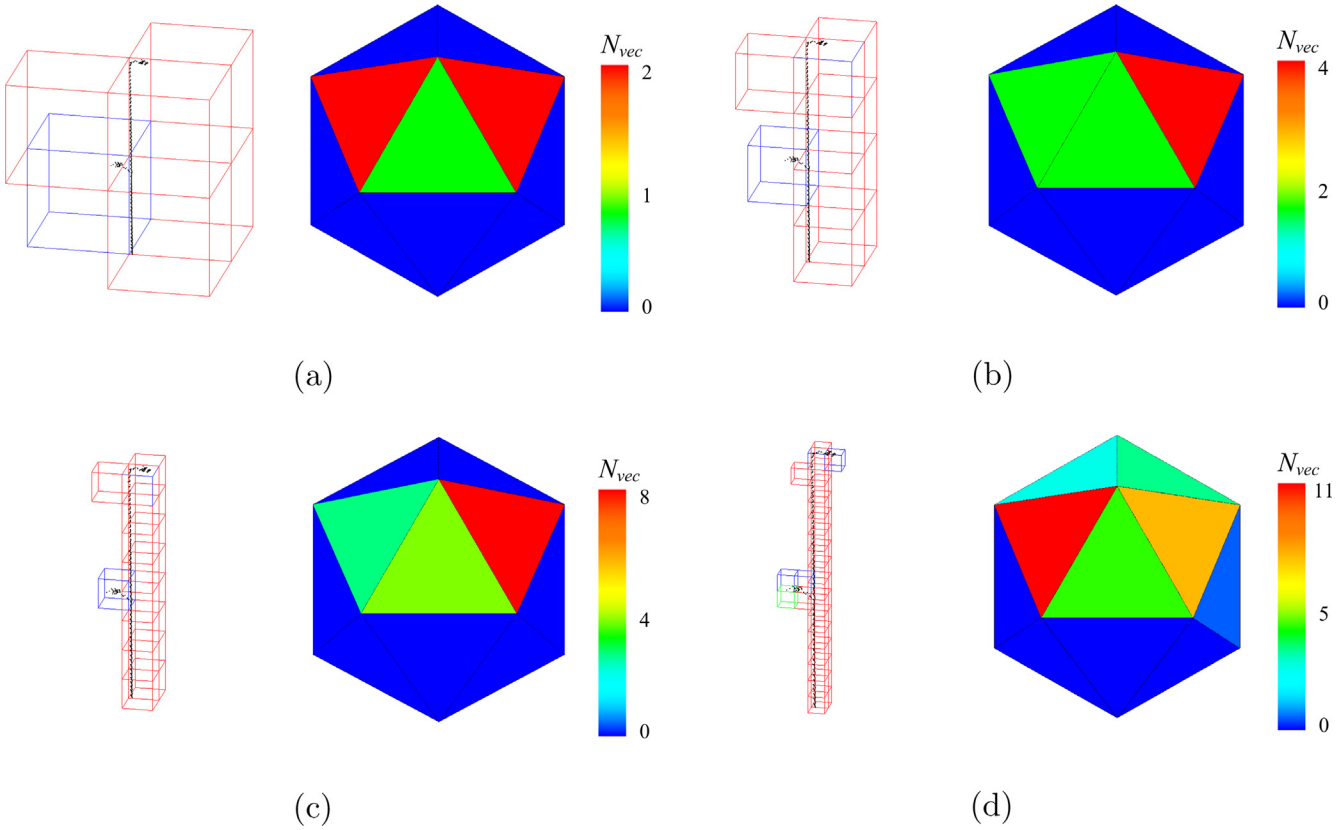

Fig. 8. Recursively subdividing a lamp pole at four levels by an octree and the corresponding *Eigen-Shpere*. (a) Subdivision at level 1. (b) Subdivision at level 2. (c) Subdivision at level 3. (d) Subdivision at level 4. In the different subdivisions, red, green and blue octants indicate linear, planar and scatter voxels respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

$$S^{c,t} = \sum_{l=1}^{n} S_l^{c,t} \tag{9}$$

Here $S^{c,t}$ is the distance between the SigVox descriptors of point cluster P_c and template cluster P_t. The preset maximum subdivision level is denoted by n.

3.5.2. Transformation of Eigen-Spheres

While determining the similarity between two *Eigen-Spheres* descriptors of the same level, i.e. the $\widehat{S}_l^{c,t}$ in Formula 8 in Section 3.5.1, the *Eigen-Sphere* of a candidate is transformed 30 times corresponding to the 30 symmetries of the icosahedron up to antipodal identification. The transformation is performed as described in Algorithm 1.

Algorithm 1. Algorithm for relatively transform the two *Eigen-Spheres* to determine similarity at subdivision level *l*.

Input: A pair of *Eigen-Spheres*, i.e. E_l^c and E_l^t.
Output: Similarity value at level *l*, i.e. $\widehat{S}_l^{c,t}$.

```
function ComputeSimilarity
  Place E_l^c and E_l^t in the standard position;
  for m = 0 → 5 do
    for n = 0 → 4 do
      Determine the (m*5 + n)-th Ŝ_l^{c,t}
      Rotate E_l^c about Z axis with 2π/5 rad;
    end for
    m = m + 1;
    Determine spherical coordinates of the m-th vertex
  of E_l^c, (1,θ_m, φ_m);
    Rotate the E_l^c about axis Z with 90 − θ_m degree;
    Rotate the E_l^c about axis X with φ degree;
  end for
  Return the minimum one among the obtained 30 Ŝ_l^{c,t};
end function
```

In Algorithm 1, the input is a pair of *Eigen-Spheres*, i.e. E_l^c and E_l^t respectively. In this step, the *Eigen-Sphere* of template cluster, denoted by E_l^t, is kept stationary and only E_l^c, which is the EGI of the candidate point cluster is transformed. The algorithm first puts the two *Eigen-Spheres* in standard position as given in Fig. 5. Note that *Vertex u* is at the positive direction of *Z* axis when the *Eigen-Sphere* is in standard position. Then the E_l^c is rotated consecutively for five times around *Z* axis by $\frac{2\pi}{5}$ rad to compute the first 5 $\widehat{S}_l^{c,t}$. The direction of rotation is performed in a right-handed manner as demonstrated in Fig. 9. This five values correspond to the five symmetries when *Vertex u* is at the north pole. The next step is to compute the five similarity values when the next vertex is at north pole position. To transform an arbitrary vertex to the north pole, for example *Vertex b*. First the spherical coordinates are computed, i.e. $(1, \theta_b, \phi_b)$, as shown in Fig. 9. Next the E_l^c is first rotated around axis *Z* by $\theta_b' = 90 - \theta_b$ degree and then rotated around *X* axis by ϕ_b degree. Subsequently *Vertex b* is transformed to the positive *Z* axis. When the next vertex is transformed to the positive *Z* axis, the 5 similarity values are computed. The algorithm runs until all the 30 $\widehat{S}_l^{c,t}$ corresponding to the 30 symmetries are all determined. Then the minimum one is obtained and returned as the similarity value of the pair of EGI descriptors and returned.

Many objects have a dominant geometric dimensionality. For example, poles are dominantly linear. For such objects, a local coordinate frame could be acquired by PCA, by aligning the object along the first eigenvector. In this way, the number of symmetries to be considered in the similarity determination can be significantly reduced. Note that in case that object has no dominant dimensionality, this possible refinement is not implemented in this work.

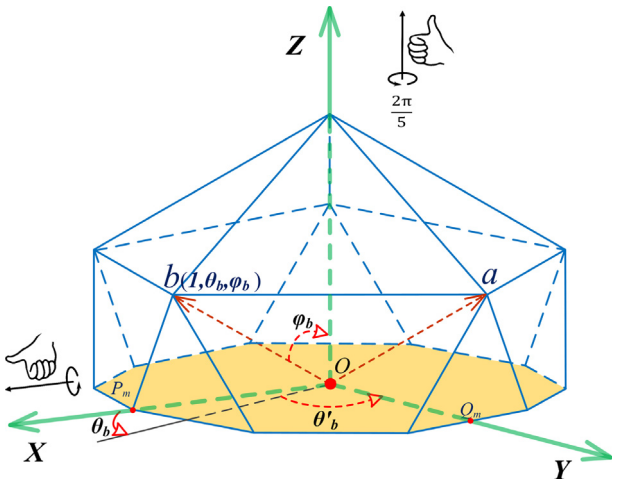

Fig. 9. Relative transformation of the E_l^c with regard to E_l^t. This figure indicates how a candidate cluster is rotated to match a training object.

3.5.3. Object recognition

This section describes the strategy of assigning the obtained candidate point clusters $\{C^i\}, i = 1, 2, \ldots, k$ to a specific kind of target object, which is represented by $\{T^j\}, j = 1, 2, \ldots, h$.

For a specific object of interest, i.e. $\{T^j\}$, its candidate point clusters are first obtained as described in Section 3.2. Then the obtained candidate point clusters, i.e. $\{C^i\}$, are subdivided by an octree into several levels. The SigVox descriptors corresponding to those levels is constructed. Next the similarity between the MS-EGI descriptors of an object of interest and candidate point clusters are determined.

To determine whether a candidate point cluster should be assigned to a specific object of interest, a preset similarity threshold is used. If the determined similarity between two SigVox descriptors, which is denoted be $S^{c,t}$ in Formula 9, is below the threshold and this similarity is minimal among different training objects, then the point cluster is assigned to the object of interest. Finally all those point clusters are exported separately.

3.6. Evaluation of the similarity

The quality of the object recognition method proposed in this work is evaluated in two ways. First, a similarity score will be determined by the similarity matching of each pair of SigVox descriptors. The similarity score denotes how confident a recognition is. The second way to evaluate the quality of object recognition is in situ inspection of the results. Finally the results are summarized in a confusion matrix where identification results are compared to ground truth.

The similarity score is computed after the similarity of the SigVox descriptors between all the candidate point clusters and a training cluster are determined. In this procedure, a confidence value will be computed that expresses the quality of the object recognition.

Suppose there are *n* levels of subdivision for a pair of SigVox descriptors. Then there are *n* preset thresholds. If among all those *n* levels of similarity, i.e. *n* pairs of *Eigen-Spheres*, there are *m* pairs

Fig. 10. Drive-Map MLS system from Fugro. (a) A side view of the MLS system. (b) A view of the sensor configuration. This system was used to acquire the point clouds used in this work.

Table 2
Specification of the Drive-Map mobile laser scanning system.

Laser pulse rate	1,333,000 p/s
Ranging accuracy	<2 cm
Maximum range	100 m
Scanner	Riegl VQ 250
Swath angle	360°
Panoramic camera	Ladybug 3

within the threshold, then this candidate point cluster will have a similarity score $F = \frac{m}{n}$. For example, suppose a point cluster is subdivided by an octree into 4 levels, then its SigVox descriptor consists of four EGI descriptors, one for each scale. When comparing the SigVox descriptors of the point cluster and its template point cluster, if 3 of them are within the thresholds, then its matching score is 0.75.

4. Results and evaluation

In this section, the proposed method is tested and validated on two MLS point clouds sampling a stretch of 4 km of urban road. First, a brief introduction of the used MLS system and a description of the used point cloud data is given. Then, the results on one of the test runs from each processing step, including pre-processing, voxelization and clustering, and object recognition, are presented and discussed. Next, the results from processing the second test run are presented. This second run samples the same road environment, but the data was acquired in opposite driving direction. This second run data was processed using the same settings. The recognition results from the two point clouds were compared and analyzed. Finally, the recognition accuracy of the proposed method was evaluated against manual in situ inspection results.

4.1. Data description

The point clouds tested in this work are acquired on March 22, 2016, by the Fugro Drive-Map MLS system, which is shown in Fig. 10. Fig. 10a shows the MLS system as a whole while Fig. 10b is a close up view of the sensors. The specifications of the MLS system are given in Table 2.

The point clouds acquired by the MLS system obtained in two opposite directions have an average point density of 1500 points per square meter. The point clouds cover a stretch of approximately 4 km of urban road. The number of points are 72,165,310 and 68,228,118 respectively. The MLS trajectory and the first point cloud are illustrated in Fig. 11. In Fig. 11a, the

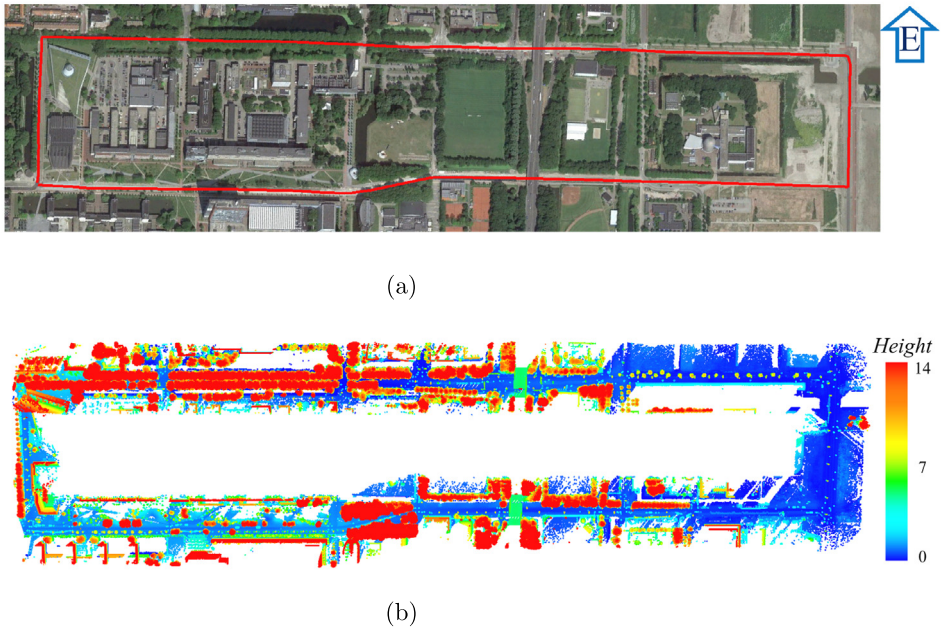

Fig. 11. Overview of the scanning trajectory and point cloud. (a) Top view of the scanning trajectory of 4 km long. (b) Original point cloud colorized by relative height.

Table 3
Parameters and thresholds used in the processing of the two point clouds.

Parameter	Value	
Tiling	Width	20 (m)
	Length	200(m)
	Overlap	5 (m)
Voxelization level		9
Dimensionality	Linearity (T_l)	10
	Planarity (T_p)	20
Bounding box ($D_{i,j}$)		5.0(%)
SigVox	Level	4
	Similarity ($S^{c,t}$)	3.0

red[1] line indicates the trajectory of the MLS system during the first acquisition. Fig. 11b denotes the original point cloud of the first acquisition colored by height.

The parameters and thresholds used in the tests are given in Table 3. Note that there are three parameters in the *Tiling* step. The *width* indicates the distance across trajectory boundary of the tiles, *length* indicates the distance along trajectory and *overlap* is the size of the buffer area between two consecutive tiles. The *Voxelization* level gives the maximum subdivision level of the octree. *Dimensionality* consists of linearity and planarity, which have dominant direction as described in Section 3.4.1. The SigVox 3D descriptor has two parameters: level indicates the number of scales, while similarity gives the threshold distance used to accept a candidate object as matching a training object.

4.2. Pre-processing of the point cloud

As the original point clouds are too large to process on a normal desktop PC and points further away from the scanning trajectory are less interesting in this study, the original point clouds are divided into smaller tiles. In the re-tiling step, points that are further than 20 m from the trajectory are removed. The length along the trajectory is 200 m for each tile with an overlap of 5 m between consecutive tiles. After re-tiling, ground and non-ground points are separated. The results of the re-tiling and separation of ground and non-ground points are given in Fig. 12.

Fig. 12a shows the bounding box of each of the 20 newly generated smaller tiles. As shown in this figure, each tile is labeled by a unique tile index. Fig. 12b shows the separation results of the 20 tiles, in which the blue and red points denote the separated ground and non-ground points respectively. Table 4 gives the number of points corresponding to each pre-processing step for both data sets. As can be noticed 33,339,127 points are left after re-tiling the first point cloud. Consecutive segmentation results in 18,562,951 ground and 14,776,176 non-ground points respectively.

After the pre-processing of the original point clouds, non-ground points are used as input for the next step, i.e. voxelization and connected component clustering.

4.3. Voxelization and clustering of non-ground points

The non-ground points of each tile are organized in an octree data structure. This section describes the results of the voxelization and clustering of the non-ground points.

[1] For interpretation of color in Fig. 11, the reader is referred to the web version of this article.

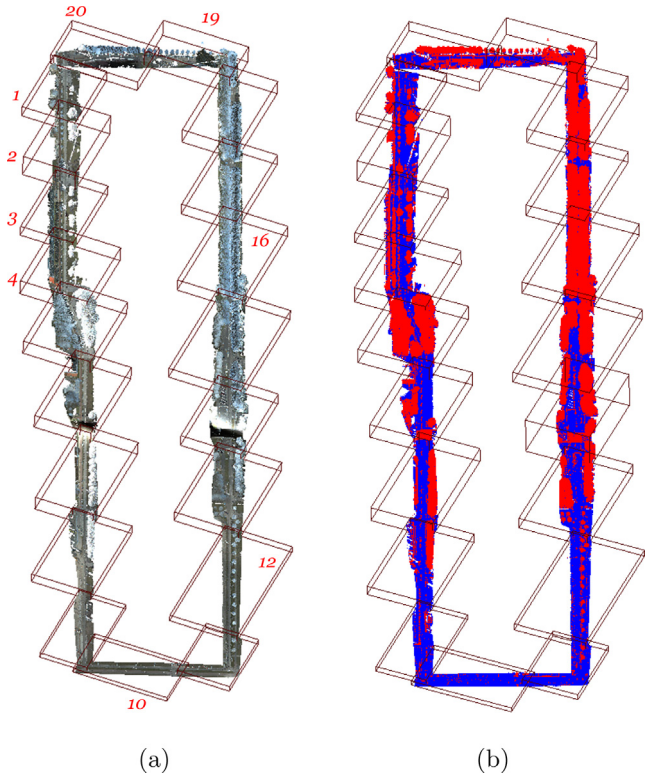

Fig. 12. Re-tiling and non-ground point separation results of the first point cloud. (a) The point cloud was divided in 20 tiles and one tile contains approximately 4 million points. (b) Separation results: ground points are plotted in blue, and non-ground points in red. During the tiling points further than 20 m from the trajectory are removed. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 4
Number of points after each pre-processing step.

Point cloud	Original	After retiling	Ground points	Non-ground points
1st Run	72,165,310	33,339,127	18,562,951	14,776,176
2nd Run	68,228,118	31,060,145	16,288,775	14,771,370

The voxelization of the non-ground points is conducted recursively in each tile, as described in Section 3.2. In this work, the criterion to stop subdivision is the minimum voxel size. The recursive subdivision of the octants in the octree is terminated whenever the voxel size is smaller than 10 cm. After voxelization, connected voxels are clustered. Consecutively, the points inside the voxels of a cluster are extracted to form a point cluster. Next, for each obtained point cluster, its 3D bounding box is obtained.

Fig. 13 shows the results of voxelization and clustering of the non-ground points from tile 3 in Fig. 12a. Fig. 13a shows the non-ground points colorized by height. Fig. 13b demonstrates an octree subdivision at level 9 corresponding to voxels of 39.4 cm. Fig. 13c illustrates the results of clustering connected voxels. The clusters are colorized by random colors. Also, the points inside the voxels of each cluster are extracted to form corresponding point clusters. The 3D bounding boxes of those extracted point clusters are given in Fig. 13d. Those 3D bounding boxes will be used to select candidate point clusters from a selected object of interest, as described in Section 3.2.

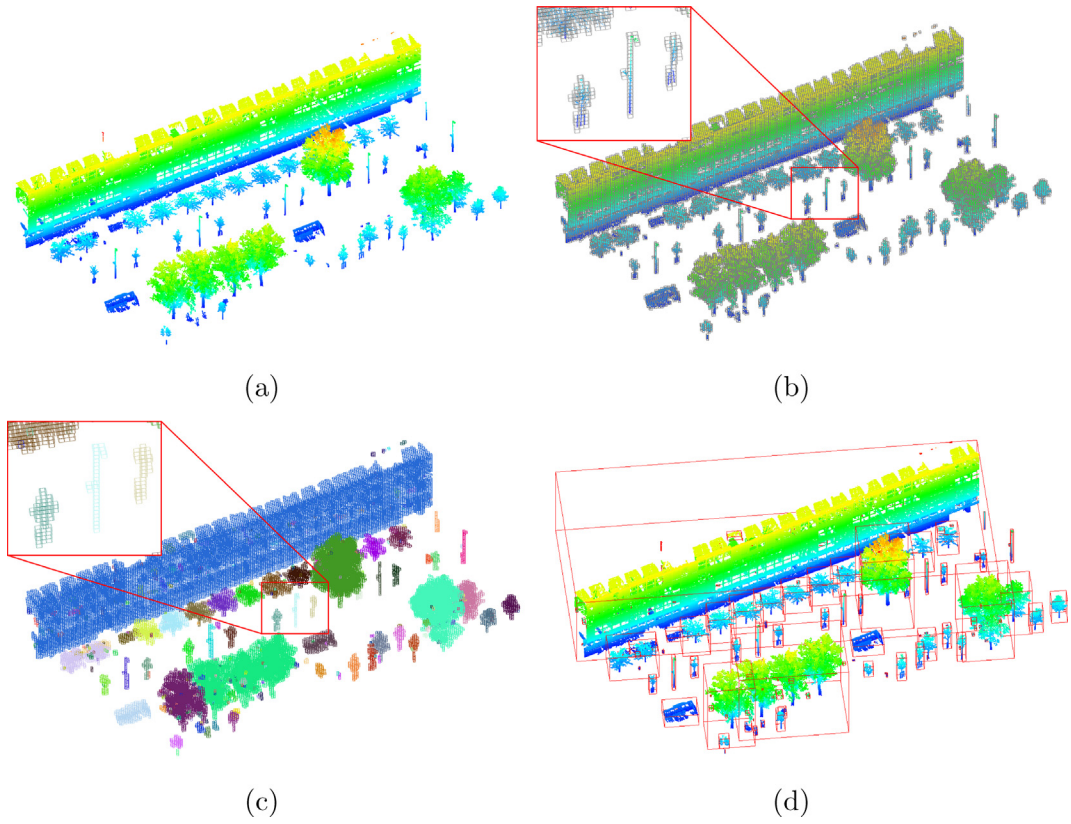

Fig. 13. Results of voxelization and clustering of a tile of non-ground points. (a) Original non-ground points of tile 3. (b) Voxelization results of the non-ground points. (c) The connected voxels are clustered. (d) 3D bounding boxes of the considered point clusters. The resulting clusters are compared with the training objects, provided their bounding boxes do not deviate too much.

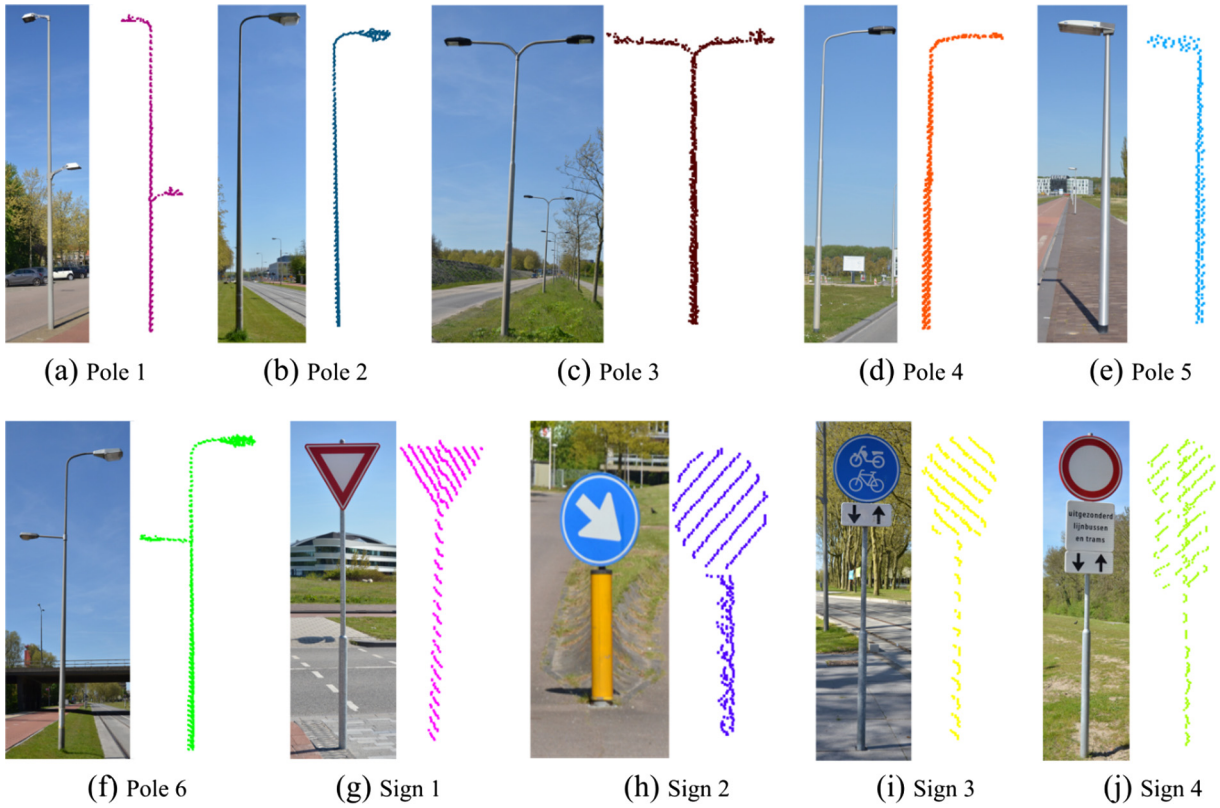

Fig. 14. Images and point clouds of the selected objects of interest. (a)–(f) are photos and points of six lamp poles, while (g)–(j) are the considered traffic signs and their corresponding point clusters. The image shows that not all objects are sampled equally well by the MLS.

Table 5
Dimensions of the 3D bounding boxes of the objects of interest.These dimensions are used to select candidate objects which speeds up processing.

Object	Bounding box (m)		
	Length	Width	Height
Pole 1	1.6	0.4	8.7
Pole 2	1.2	0.8	7.7
Pole 3	3.5	0.5	7.5
Pole 4	1.9	0.5	9.6
Pole 5	0.7	0.3	3.4
Pole 6	3.4	0.4	9.7
Sign 1	0.9	0.1	2.9
Sign 2	0.4	0.2	1.2
Sign 3	0.6	0.2	2.5
Sign 4	0.7	0.2	3.1

4.4. Object recognition

Results of object recognition are presented in this section. In this work, 6 different street lamp poles and 4 different traffic signs were selected as objects of interest. Points corresponding to example objects were manually extracted as template point clusters and object recognition was conducted on the rest of the point cloud. Fig. 14 illustrates the selected objects. In the figure, the top row are photos of the selected street lap poles and their sampled point clusters, while the bottom row shows the considered road signs and their corresponding point clusters.

After importing the point clusters of the selected objects of interest, their 3D bounding boxes are obtained, as given in Table 5. Then, for candidate object selection, the 3D bounding box of each object of interest is compared with that of the voxel clusters obtained in Section 4.3. In this work, the threshold for the similarity of 3D bounding boxes ($D_{i,j}$) is set to 5.0%. Take *Pole 2* and *Pole 4* in Fig. 14 for example, the distance of their bounding boxes calculated from Eq. (1) is 4.4%. Thus, not only the poles of type *Pole 2* in the test data will be selected as candidates, but also the poles of type *Pole 4*. In point cloud of the first run, there are 37 and 5 poles of type *Pole 2* and *Pole 4* are obtained as candidates of *Pole 2*.

In the next step, the SigVox 3D descriptors of *Pole 2* and all the 42 selected candidate point clusters will be compared. Fig. 15 shows the subdivision at 4 levels of *Pole 2* and *Pole 4* and the corresponding SigVox descriptors. Next, the similarity distances of between SigVox descriptors of the training point cluster of *Pole 2* and the obtained 42 candidates are determined. The resulting similarity distances for each of the 4 subdivision levels are shown in Fig. 16. The histograms show that there is a clear difference between the similarity distance *Pole 2* and *Pole 4*. In this work, the similarity distance threshold is set to 3.0. Thus only candidates that have a similarity distance below 3.0 will be assigned to type *Pole 2*. In the point cloud of the first run, 37 poles of type *Pole 2* are correctly identified. This procedure is performed for all selected objects of interest.

For better visualization, the object recognition results are presented in two figures: Figs. 17 and 18. The object recognition results of the north part of the study area from the first point cloud are given in Fig. 17. In Fig. 17a, the green icons denote the correctly recognized objects. The red icons depict items that are not correctly identified. In Fig. 17b, ground points are colored light blue and non-ground points gray. The successfully recognized objects are colored in correspondence to Fig. 14. Fig. 17b shows a scenario

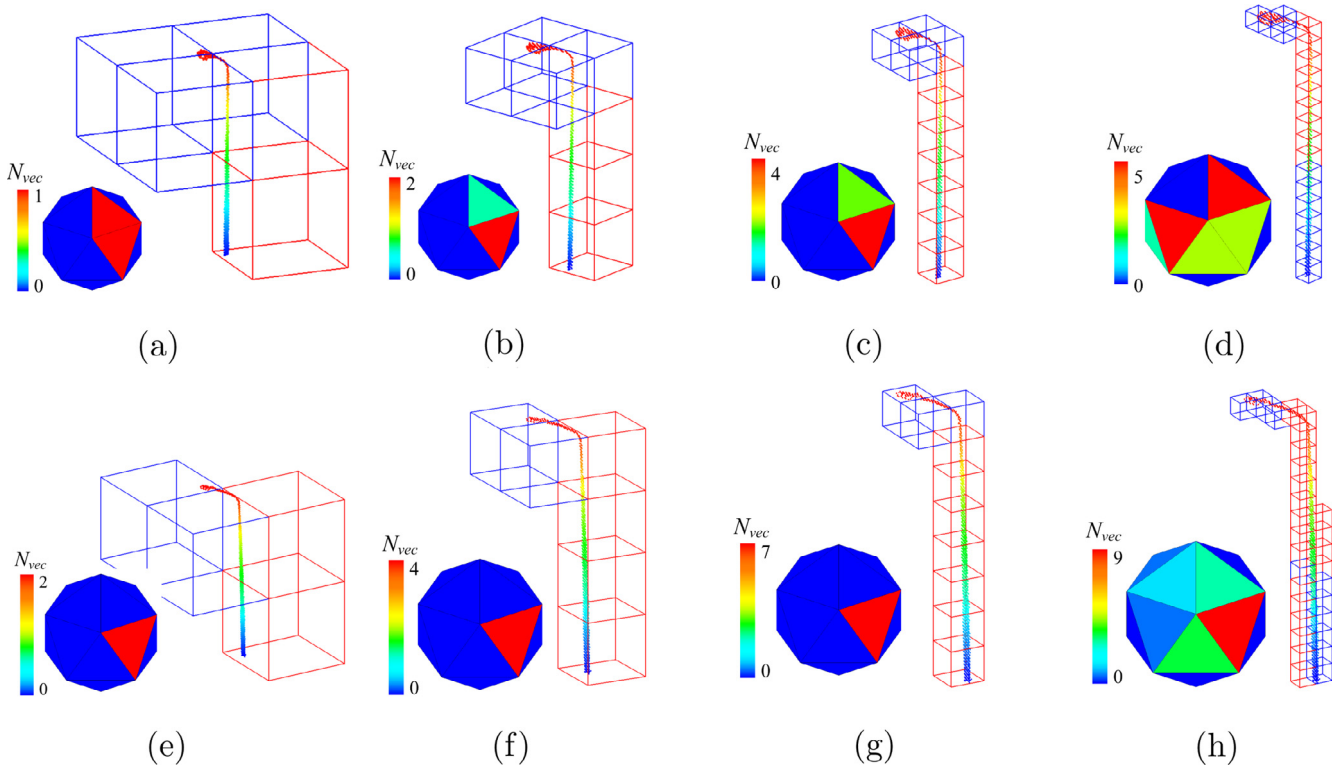

Fig. 15. The voxel subdivision and the correspondent SigVox 3D feature descriptor of *Pole 2* and *Pole 4*of the four levels. (a)–(d) are the voxel subdivision and SigVox feature descriptor of *pole 2* at level 1, 2, 3 and 4. (e)–(h) are the voxel subdivision and SigVox feature descriptor of *Pole 4* at level 1, 2, 3 and 4. The red, green and blue voxel indicate the linearity, planarity and scatter of the points inside each voxel. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

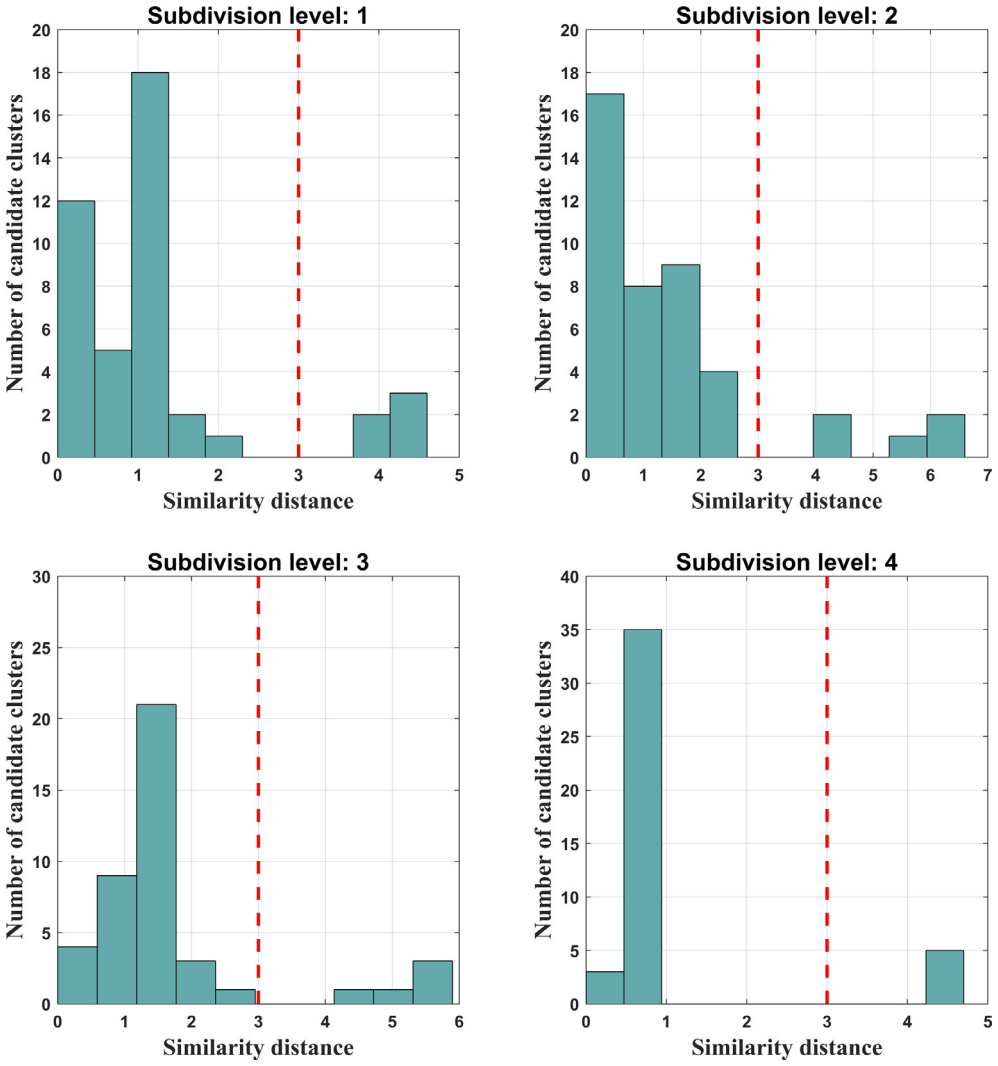

Fig. 16. Similarity distances of the SigVox descriptor between template *Pole 2* to the obtained 42 candidates at 4 levels of subdivision. The red dash line in each subfigure denotes the threshold of similarity distance. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

that has three lamp poles of type *Pole 1* in Fig. 14. The proposed method identified two of them. The pole in black was missed because it is close to a bus stop. Therefore its points were in a cluster together with the bus stop points. As a result, the bounding box of this cluster was too far away from the bounding boxes of the training objects. Two road signs, of type *Sign 3* and *Sign 4* in Fig. 14, which are highlighted by rectangles, are correctly recognized in Fig. 17b. Fig. 17c shows a scenario where seven lamp poles of type *Pole 1* and four road signs of type *Sign 3* and *Sign 4* were correctly recognized. In Fig. 17d, there are actually four lamp poles of type *Pole 4*. However, one lamp pole, the black one, was not recognized because it is connected with the overhead roadside tree and was therefore not selected as candidate. In Fig. 17e, four street lamps of type *Pole 4* and 5 road signs of type *Sign 2* and *Sign 3* were all correctly recognized.

Fig. 18 shows the recognition results from the south part of the study area. Fig. 18a is a top view of the area. Fig. 18b is a zoom in of Zone E in Fig. 18a. Three poles of type *Pole 4* and *Pole 5* were successfully recognized. Also, two road signs of type *Sign 1* and one road sign of type *Sign 3* were correctly identified. However, one road sign of type *Sign 1* was not identified. This is because as a result of occlusion, only part of its shape is represented by the available points. In Fig. 18c two lamp poles of type *Pole 3* and *Pole 4* were successfully identified. Four road signs of type *Sign 3* were correctly recognized. Also one sign of type *Sign 1* and one of type *Sign 2* were identified successfully.

4.5. Evaluation of object recognition results

To validate the reliability and accuracy of the proposed road object recognition method, a second point cloud of the same area was collected, but from an opposite driving direction. This point cloud was processed by the same work flow. The ground truth of the street lamp poles and road signs was also collected by visual in situ inspection. The recognition results of the second point cloud and ground truth of the road objects are given in Table 6.

As illustrated in Table 6, 123 and 125 from a total of 130 street lamp poles are correctly recognized in the two point clouds. Thus the accuracy of street lamp poles recognition rates are 94% and 96% for the first and the second point cloud. There are 47 and 48 road signs correctly recognized from a total of 51 road signs. Therefore the accuracy of road sign recognition for the two point clouds is 92% and 94%. The overall accuracy of the road structure recognition are 94% and 96% for the two point clouds.

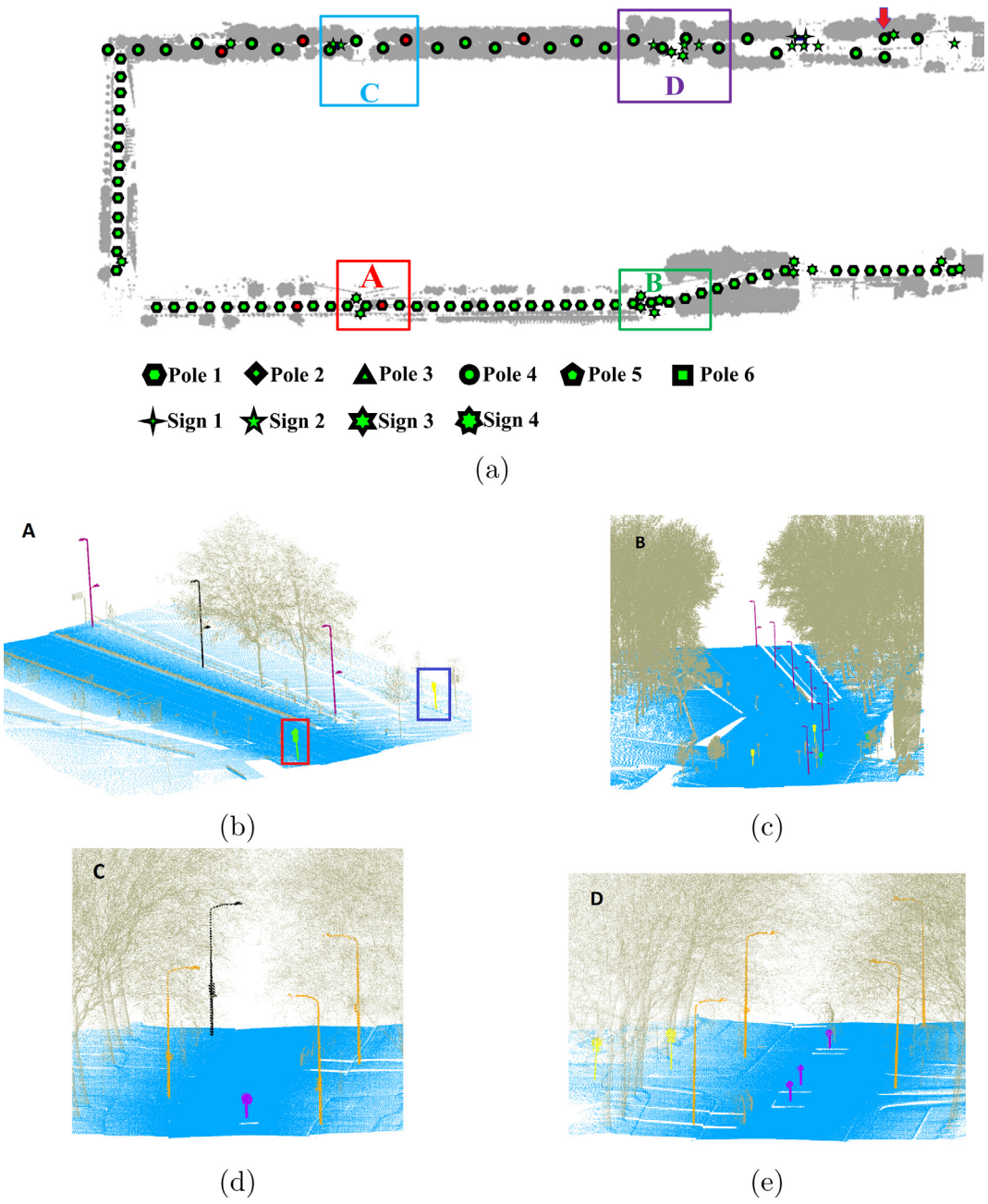

Fig. 17. Street object recognition results from the north part of the study area. Different icons indicate different lamp pole and traffic sign types. Successfully identified objects are indicated in green, missed objects are colored red. (a) Overall results of the recognition. (b) Zoomed in view of area A. (c) Zoomed in view of area B. (c) Zoomed in view of area C. (d) Zoomed in view of area D. Each object type is colored in a different color. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

There are a few cases where poles are correctly identified in the point cloud of the first run, but missed in the second run. These inconsistencies are caused either by occlusions, or by a too large distance of an object to the scanning trajectory. These factors result in incomplete sampling of the objects and resulting in deviating 3D bounding box sized. As a result their clusters were not selected as a candidate object. An example is the road sign of type *Sign 1* in Fig. 18b. However, if the full shape of an object is sampled, still the object can be successfully identified even if there exists a big difference in point density. Notably, the proposed method is able to identify poles sampled at different point density. Fig. 19a shows the lamp pole denoted by the red arrow in Fig. 17a. As the distances of the pole to the scanning trajectory are different, the sampled point density is different as well. Figs. 19b and c are the point clouds of the pole sampled from two opposite driving directions, consisting of 954 and 273 points respectively. There is a big difference in point density at the top of the pole as indicated in the figures. Still, the pole was correctly recognized in both point clouds.

Recognition may fail if an object is too close to another object. For example, Fig. 20 shows a scenario where a lamp pole is connected to a road side tree. As a consequence the lamp pole is not separated in the clustering step. Subsequently it was not

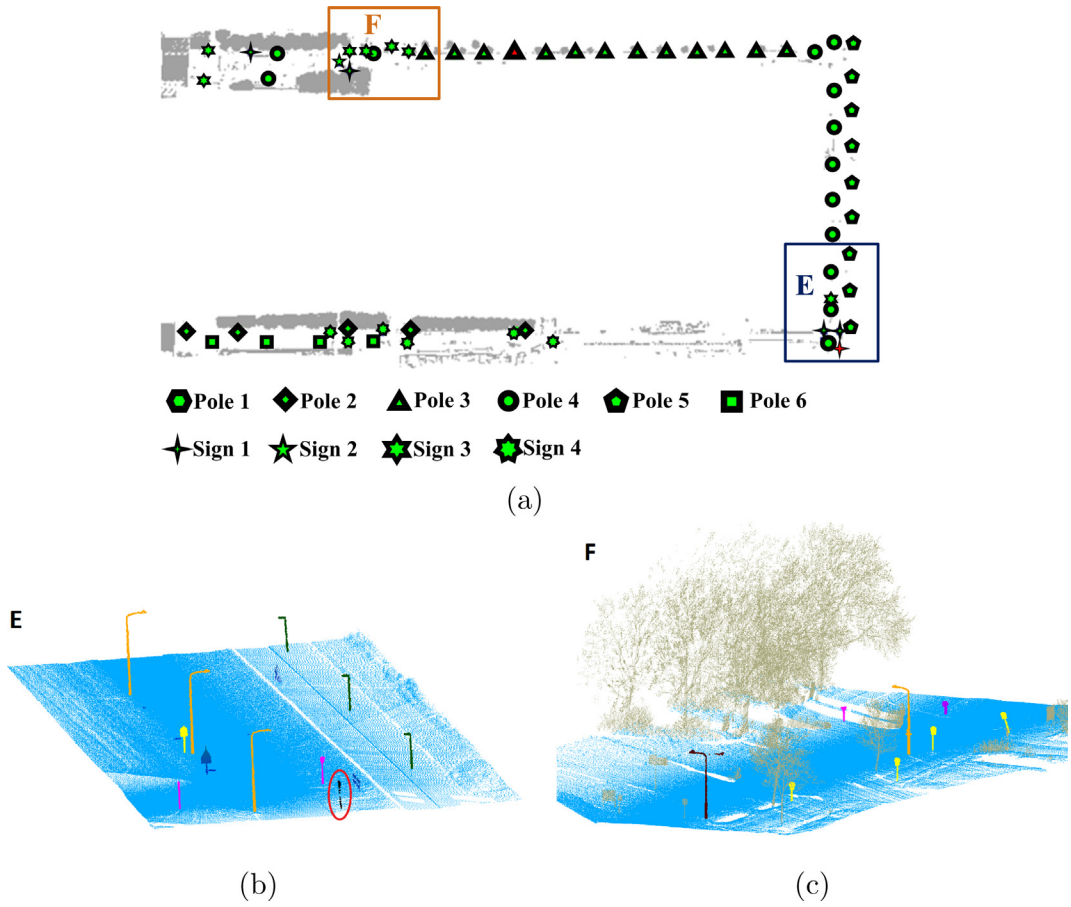

Fig. 18. Street object recognition results from the south part of the study area. (a) Overall results of the recognition in the south part of the study area. (b) Zoomed in view of area E. (c) Zoomed in view of area F. The red ellipse indicates a street sign that was not detected by the workflow. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 6
Results of road object recognition from the two point clouds and in situ inspected ground truth.

Object	Pole 1	1st Point cloud	2nd Point cloud	Ground truth	Ground truth total
Street lamp pole	Pole 1	56	56	58	130
	Pole 2	37	39	41	
	Pole 3	12	12	12	
	Pole 4	5	5	5	
	Pole 5	9	9	9	
	Pole 6	4	4	4	
Road sign	Sign 1	6	5	7	51
	Sign 2	10	9	10	
	Sign 3	15	16	16	
	Sign 4	16	18	18	
Object total		170	173	181	181

considered in the candidate selection step and finally not recognized. Further work should consider the separation of apparent connected objects.

5. Discussion and conclusions

5.1. Discussion

In this section, the sensitivity analysis of the used parameters is firstly given. Then, future work on some aspects of the proposed method are discussed.

5.1.1. Sensitivity analysis

The parameters used in this work are given in Table 3. This section gives a short analysis of the influence of the parameters to the results.

1. Voxelization level. The level of voxelization corresponds to the size of the voxels, the deeper the level, the smaller the voxel size. The parameter should be set considering the minimum distance between objects of interest and the surroundings, as well as the average point density.
2. Dimensionality. The linearity and planarity thresholds will define the dimensionality of a voxel using PCA. The smaller

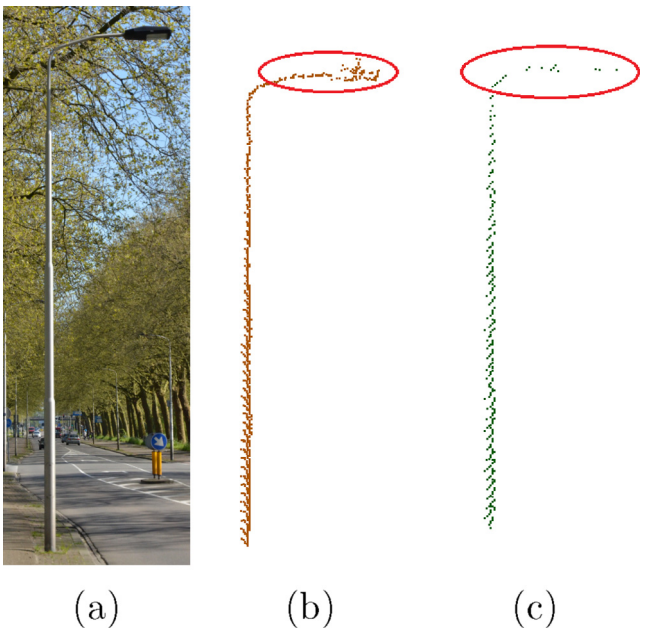

Fig. 19. A lamp pole of type *Pole 4* represented in two point clouds obtained from opposite driving directions. (a) A photo of the pole. (b) Points from the first point cloud. (c) Points from the second point cloud. In the second run, the pole was sampled by 273 points compared to 954 points in the first run, which strongly affects the local point density as visible in the area marked by the ellipse.

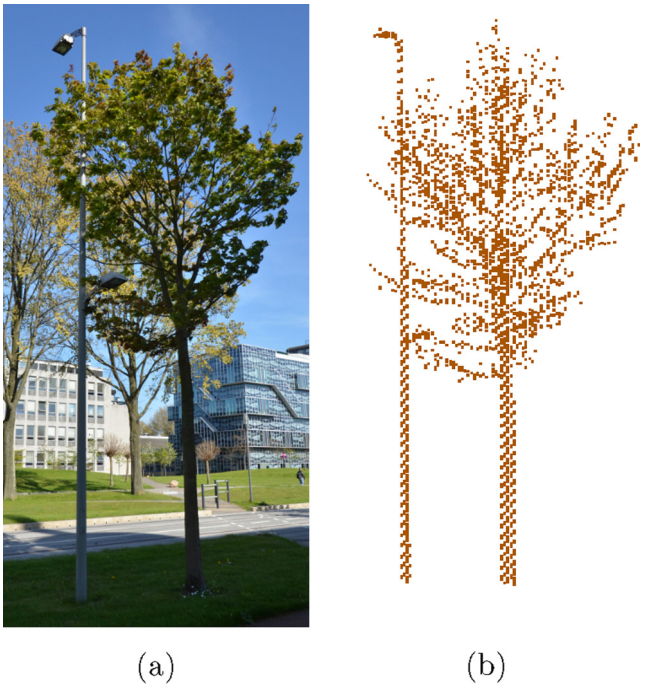

Fig. 20. A scenario where a lamp pole was not identified. Because the tree and the pole are too close, they are clustered together, which negatively effects the shape encoding. (a) A lamp pole connected to a road side tree. (b) Point cloud of the lamp pole and the tree.

the thresholds, the more significant voxels will contribute their eigenvectors to the SigVox descriptor. The optimal thresholds should also consider the noise level of the point clouds.
3. Bounding box. This parameter is used to remove point clusters that have a deviating 3D bounding box compared to those of the training objects at an early stage. This avoids considering all the obtained point clusters and speeds up the processing. However, a too small bounding box buffer will cause omissions.
4. SigVox. The number of required levels of the SigVox descriptor depends on the complexity of the geometric shape of the selected objects of interest. A too small number of levels will result in robustness issues. The similarity distance threshold depends on the similarity of the considered objects. Smaller thresholds may leads to omissions in the recognition results.

5.1.2. Future work

Still some aspects of the presented method should either be further considered or improved.

1. Object separation. As the results in Section 4.4 shows, objects were unsuccessfully recognized because of their proximity to other objects. To enhance performance, separation of objects of interest needs to be further improved.
2. Candidate selection. In this work, candidate point clusters are selected by comparing their 3D bounding boxes with the 3D bounding boxes of the example objects of interest. However, there may be situations in which poles are inclined. Such cases are notably interesting for street inventory management. Probably the method proposed in this work will not identify inclined street poles, because of the initial bounding box selection step.
3. Approximation of EGI. The EGI descriptor is approximated by an icosahedron, which has 20 boundary triangles. If the shape of an object is extremely complicated, the icosahedron may not be sufficient for representing the shape of the object. However, the icosahedron can be further tessellated by incrementally subdividing one triangle into four smaller triangles until the approximation meets the requirement.
4. Threshold selection. The used parameters are mainly set with regard to the experimental results in this work. A next work should consider automatic threshold selection.

5.2. Conclusions

In this work, an automatic method for roadside furniture identification is proposed and validated. The method consists of four steps, i.e. pre-processing, voxelization and SigVox descriptor construction, template matching, and result validation. The proposed method was tested on two point clouds sampling the same stretch of 4 km of urban road obtained by a MLS system driving in opposite direction. In this study, 6 different types of street lamp poles and 4 types of road signs were selected as objects of interest and the SigVox descriptor of those objects were constructed as template objects. The recognition was performed by computing the distance between the SigVox descriptors of template objects and candidate point clusters. The recognition results of the two point clouds were compared to ground truth data of the street objects obtained by in situ visual inspection. The comparison results show that the overall accuracy of road structure recognition is 94% and 96% for the two point clouds. To the best of our knowledge, this is the first time that a shape descriptor, describing complete objects are used to efficiently extract repetitive objects in large point cloud scenes.

Acknowledgments

The authors would like to thank the anonymous reviewers very much for reviewing this manuscript. The authors would like to thank the Chinese Scholarship Council and the IQmulus project (FP7-ICT-2011-318787) for supporting this research. Also, the authors would like to thank Martin Kodde and Sjoerd Staats from Fugro GeoServices B.V. for the MLS point clouds used in this work.

References

Agamennoni, G., Nieto, J.I., Nebot, E.M., 2011. Robust inference of principal road paths for intelligent transportation systems. IEEE Trans. Intell. Transport. Syst. 12 (1), 298–308.

Allen, K., 1971. Patterns and search statistics. In: Rustagi, J.S. (Ed.), Optimizing Methods in Statistics. Academic Press, pp. 303–337.

Babahajiani, P., Fan, L., Gabbouj, M., 2015. Object recognition in 3d point cloud of urban street scene. Computer Vision - ACCV 2014 Workshops 0302-9743, November, vol. 9008. Springer International Publishing, Singapore, pp. 177–190.

Barber, D., Mills, J., Smith-Voysey, S., 2008. Geometric validation of a ground-based mobile laser scanning system. ISPRS J. Photogram. Rem. Sens. 63 (1), 128–141.

Batty, M., Axhausen, K.W., Giannotti, F., Pozdnoukhov, A., Bazzani, A., Wachowicz, M., Ouzounis, G., Portugali, Y., 2012. Smart cities of the future. Euro. Phys. J.: Spec. Top. 214 (1), 481–518.

Bishop, R., 2000. A survey of intelligent vehicle applications worldwide. In: Proceedings of the IEEE Intelligent Vehicles Symposium 2000 (Cat. No. 00TH8511), August, pp. 25–30.

Bremer, M., Wichmann, V., Rutzinger, M., 2013. eigen value and graph-based object extraction from mobile laser scanning point clouds. In: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. Vol. II-5/W2. ISPRS Workshop Laser Scanning 2013. Antalya, Turkey, November 11–13, pp. 55–60.

Brenner, C., 2009. Extraction of features from mobile laser scanning data for future driver assistance systems. In: Lecture Notes in Geoinformation and Cartography, pp. 25–42.

Cabo, C., Ordoñez, C., García-Cortés, S., Martínez, J., 2014. An algorithm for automatic detection of pole-like street furniture objects from Mobile Laser Scanner point clouds. ISPRS J. Photogram. Rem. Sens. 87, 47–56.

Cahalane, C., McCarthy, T., McElhinney, C., 2010. Mobile mapping system performance - an initial investigation into the effect of vehicle speed on laser scan lines. In: Proceedings of Remote Sensing and Photogrammetry Society Annual Conference, pp. 1–8.

Conway, J., Burgiel, H., Goodman-Strauss, C., 2008. The Symmetries of Things. Ak Peters Series. Taylor and Francis.

Díaz-Vilarino, L., González-Jorge, H., Martínez-Sánchez, J., Bueno, M., Arias, P., 2016. Determining the limits of unmanned aerial photogrammetry for the evaluation of road runoff. Measurement 85, 132–141.

El-Halawany, S., Lichti, D., 2011. Detection of road poles from mobile terrestrial laser scanner point cloud. In: 2011 International Workshop on Multi-Platform/Multi-Sensor Remote Sensing and Mapping (M2RSM), January, pp. 1–6.

Ellum, C., El-Sheimy, N., 2002. Land-based mobile mapping systems. Photogram. Eng. Rem. Sens. 68 (1), 15–17.

Fan, H., Yao, W., Tang, L., 2014. Identifying man-made objects along urban road corridors from mobile lidar data. IEEE Geosci. Rem. Sens. Lett. 11 (5), 950–954.

Frueh, C., Zakhor, A., 2003. Constructing 3D city models by merging ground-based and airborne views. In: Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003, vol. 2, pp. II–562–9.

Golovinskiy, A., Kim, V., Funkhouser, T., 2009. Shape-based recognition of 3D point clouds in urban environments. In: International Conference on Computer Vision (ICCV), pp. 2154–2161.

Guan, H., Li, J., Yu, Y., Wang, C., Chapman, M., Yang, B., 2014. Using mobile laser scanning data for automated extraction of road markings. ISPRS J. Photogram. Rem. Sens. 87, 93–107.

Haala, N., Brenner, C., 1999. Extraction of buildings and trees in urban environments. ISPRS J. Photogram. Rem. Sens. 54 (2-3), 130–137.

Haala, N., Peter, M., Kremer, J., Hunter, G., 2008. Mobile lidar mapping for 3D point cloud collection in urban areas: a performance test. Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci. XXXVII-B5, 1119–1124.

Halfawy, M., 2008. Integration of municipal infrastructure asset management processes: challenges and solutions. J. Comput. Civil Eng. 22 (3), 216–229.

Horn, B., 1984. Extended Gaussian Images. Proc. IEEE 72 (12), 1671–1686.

Ivan, I., Benenson, I., Jiang, B., Horák, J., Haworth, J., Inspektor, T., 2015. Geoinformatics for intelligent transportation. Lecture Notes in Geoinformation and Cartography, vol. 214, p. 272.

Jaakkola, A., Hyyppä, J., Hyyppä, H., Kukko, A., 2008. Retrieval algorithms for road surface modelling using laser-based mobile mapping. Sensors 8 (9), 5238–5249.

Kumar, P., McElhinney, C., Lewis, P., McCarthy, T., 2014. Automated road markings extraction from mobile laser scanning data. Int. J. Appl. Earth Observ. Geoinf. 32, 125–137.

Lehtomäki, M., Jaakkola, A., Hyyppä, J., Kukko, A., Kaartinen, H., 2010. Detection of vertical pole-like objects in a road environment using vehicle-based laser scanning data. Rem. Sens. 2 (3), 641–664.

Li, D., Oude Elberink, S., 2013. Optimizing detection of road furniture (pole-like object) in mobile laser scanner data. In: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. Vol. II-5/W2. ISPRS Workshop Laser Scanning 2013, Antalya, Turkey, November 11–13, pp. 55–60.

Li, Q., Zheng, N., Cheng, H., 2004. Springrobot: a prototype autonomous vehicle and its algorithms for lane detection. IEEE Trans. Intell. Transport. Syst. 5 (4), 300–308.

Lindenbergh, R., Berthold, D., Sirmacek, B., Herrero-Huerta, M., Wang, J., Ebersbach, D., 2015. Automated large scale parameter extraction of road-side trees sampled by a laser mobile mapping system. In: Remote Sensing and Spatial Information Sciences. No. XL-3 in International Archives of the Photogrammetry. ISPRS, La Grande Motte, France, 28 September–3 Ocotober, pp. 589–594.

Mc Elhinney, C., Kumar, P., Cahalane, C., McCarthy, T., 2010. Initial results from European Road Safety Inspection (EURSI) mobile mapping project. ISPRS Commission V Technical Symposium, vol. 2007, pp. 440–445.

Nebiker, S., Bleisch, S., Christen, M., 2010. Rich point clouds in virtual globes - a new paradigm in city modeling? Comp., Environ. Urban Syst. 34 (6), 508–517.

Over, M., Schilling, A., Neubauer, S., Zipf, A., 2010. Generating web-based 3D City Models from OpenStreetMap: the current situation in Germany. Comp., Environ. Urban Syst. 34 (6), 496–507.

Payeur, P., 2006. A computational technique for free space localization in 3-D multiresolution probabilistic environment models. IEEE Trans. Instrument. Measur. 55, 1734–1746.

Pfeifer, N., 2001. Derivation of digital terrain models in the Scop++ environment. In: OEEPE Workshop on Airborne Laserscanning and Interferometric SAR for Digital Elevation Models, vol. 13.

Preparata, F., Shamos, M., 1985. Computational Geometry: An Introduction. Springer-Verlag, New York.

Pu, S., Rutzinger, M., Vosselman, G., Oude Elberink, S., 2011. Recognizing basic structures from mobile laser scanning data for road inventory studies. ISPRS J. Photogram. Rem. Sens. 66 (6).

Pu, S., Zhan, Q., 2009. Classification of mobile terrestrial laser point clouds using semantic constraints. In: SPIE Optical Engineering and Applications. 74470D1–74470D9.

Puente, I., González-Jorge, H., Martínez-Sánchez, J., Arias, P., 2013. Review of mobile mapping and surveying technologies. Measurement 46 (7), 2127–2145.

Puttonen, E., Jaakkola, A., Litkey, P., Hyyppä, J., 2011. Tree classification with fused mobile laser scanning and hyperspectral data. Sensors 11 (5), 5158–5182.

Qin, R., Gruen, A., 2014. 3D change detection at street level using mobile laser scanning point clouds and terrestrial images. ISPRS J. Photogram. Rem. Sens. 90, 23–35.

Rodríguez-Cuenca, B., García-Cortés, S., Ordóñez, C., Alonso, M., 2015. Automatic detection and classification of pole-like objects in urban point cloud data using an anomaly detection algorithm. Rem. Sens. 7 (10), 12680–12703.

Rusu, R., Blodow, N., Beetz, M., 2009. Fast Point Feature Histograms (FPFH) for 3D registration. In: IEEE International Conference on Robotics and Automation, pp. 3212–3217.

Rutzinger, M., Pratihast, A., Elberink, O., Vosselman, G., 2010. Detection and modelling of 3D trees from mobile laser scanning data. Int. Arch. Photogram., Rem. Sens. Spat. Inf. Sci. 38 (5), 520–525.

Schreiber, M., Knöppel, C., Franke, U., 2013. Laneloc: Lane marking based localization using highly accurate maps. In: 2013 IEEE Intelligent Vehicles Symposium (IV), June, pp. 449–454.

Teo, T.A., Chiu, C.M., 2015. Pole-like road object detection from mobile lidar system using a coarse-to-fine approach. IEEE J. Select. Top. Appl. Earth Observ. Rem. Sens. 8 (10), 4805–4818.

Vanier, D., 2006. Towards sustainable municipal infrastructure asset management. Handb. Urban Sustain. 12 (1), 283–314.

Velizhev, A., Shapovalov, R., Schindler, K., 2012. Implicit shape models for object detection in 3D point clouds. XXII ISPRS Congress, Technical Commission III, vol. I-3, pp. 179–184.

Vosselman, G., Maas, H., 2010. Airborne and Terrestrial Laser Scanning. Whittles Publishing.

Wang, J., González-Jorge, H., Lindenbergh, R., Arias-Sánchez, P., Menenti, M., 2013. Automatic estimation of excavation volume from laser mobile mapping data for mountain road widening. Rem. Sens. 5 (9), 4629–4651.

Wang, J., González-Jorge, H., Lindenbergh, R., Arias-Sánchez, P., Menenti, M., 2014. Geometric road runoff estimation from laser mobile mapping data. ISPRS Annals of the Photogrammetry, Riva del Garda, Italy, June, vol. II-5, pp. 385–391.

Wang, J., Lindenbergh, R., Shen, Y., Menenti, M., 2016. Coarse point cloud registration by EGI matching of voxel clusters. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. XXIII ISPRS Congress, Prague, Czech Republic, July, vol. III-5, pp. 98–103.

Wu, B., Yu, B., Yue, W., Shu, S., Tan, W., Hu, C., Huang, Y., Wu, J., Liu, H., 2013. A voxel-based method for automated identification and morphological parameters estimation of individual street trees from mobile laser scanning data. Rem. Sens. 5, 584–611.

Xiao, W., Vallet, B., Schindler, K., Paparoditis, N., 2016. Street-side vehicle detection, classification and change detection using mobile laser scanning data. ISPRS J. Photogram. Rem. Sens. 114, 166–178.

Yang, B., Dong, Z., 2013. A shape-based segmentation method for mobile laser scanning point clouds. ISPRS J. Photogram. Rem. Sens. 81, 19–30.

Yang, B., Dong, Z., Zhao, G., Dai, W., 2015. Hierarchical extraction of urban objects from mobile laser scanning data. ISPRS J. Photogram. Rem. Sens. 99, 45–57.

Yang, B., Fang, L., Li, J., 2013. Semi-automated extraction and delineation of 3D roads of street scene from mobile laser scanning point clouds. ISPRS J. Photogram. Rem. Sens. 79, 80–93.

Yang, B., Fang, L., Li, Q., Li, J., 2012. Automated extraction of road markings from mobile lidar point clouds. Photogram. Eng. Rem. Sens. 78 (4), 331–338.
Yu, W., He, F., Xi, P., 2010. A rapid 3D seed-filling algorithm based on scan slice. Comp. Graph. 34 (4), 449–459.
Yu, Y., Li, J., Guan, H., Wang, C., Yu, J., 2015. Semiautomated extraction of street light poles from mobile LiDAR point-clouds. IEEE Trans. Geosci. Rem. Sens. 53 (3), 1374–1386.
Zhong, R., Wei, J., Su, W., Chen, Y., 2013. A method for extracting trees from vehicle-borne laser scanning data. Math. Comp. Model. 58 (3–4), 727–736.
Zhou, L., Vosselman, G., 2012. Mapping curbstones in airborne and mobile laser scanning data. Int. J. Appl. Earth Observ. Geoinf. 18 (1), 293–304.