ISPRS Journal of Photogrammetry and Remote Sensing 150 (2019) 44–58

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

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

Learning high-level features by fusing multi-view representation of MLS point clouds for 3D object recognition in road environments[☆]

Zhipeng Luo[a], Jonathan Li[a,b,*], Zhenlong Xiao[a], Z. Geroge Mou[a], Xiaojie Cai[a], Cheng Wang[a]

[a] *Fujian Key Laboratory of Sensing and Computing for Smart Cities, School of Information Science and Engineering, Xiamen University, 422 Siming Road South, Xiamen, FJ 361005, China*

[b] *Departments of Geography and Environmental Management and Systems Design Engineering, University of Waterloo, 200 University Avenue West, Waterloo, ON N2L 3G1, Canada*

A R T I C L E I N F O

Keywords:
Convolutional neural networks
3D object recognition
MLS point clouds
Multi-view representation
Two-stage fusion network

A B S T R A C T

Most existing 3D object recognition methods still suffer from low descriptiveness and weak robustness although remarkable progress has made in 3D computer vision. The major challenge lies in effectively mining high-level 3D shape features. This paper presents a high-level feature learning framework for 3D object recognition through fusing multiple 2D representations of point clouds. The framework has two key components: (1) three discriminative low-level 3D shape descriptors for obtaining multi-view 2D representation of 3D point clouds. These descriptors preserve both local and global spatial relationships of points from different perspectives and build a bridge between 3D point clouds and 2D Convolutional Neural Networks (CNN). (2) A two-stage fusion network, which consists of a deep feature learning module and two fusion modules, for extracting and fusing high-level features. The proposed method was tested on three datasets, one of which is Sydney Urban Objects dataset and the other two were acquired by a mobile laser scanning (MLS) system along urban roads. The results obtained from comprehensive experiments demonstrated that our method is superior to the state-of-the-art methods in descriptiveness, robustness and efficiency. Our method achieves high recognition rates of 94.6%, 93.1% and 74.9% on the above three datasets, respectively.

1. Introduction

The Mobile Laser Scanning (MLS) system is increasingly chosen for autonomous vehicles (AVs) or intelligent transportation systems (ITS) (Broggi et al., 2013; Schreiber et al., 2013; Seo et al., 2015), because it collects dense and accurate 3D point clouds efficiently from large areas. As a result, in recent years, 3D point clouds have been widely used in various related applications, such as 3D object detection in roadways (Guan et al., 2014; Yang et al., 2015; Yu et al., 2016; Wen et al., 2016; Yang et al., 2017; Yu et al., 2017), object modeling and 3D reconstruction (Shah et al., 2017; Hu et al., 2018), semantic segmentation (Engelmann et al., 2017; Dong et al., 2018), and registration (Yu et al., 2015a; Zai et al., 2017). As a vital part of preprocessing steps, 3D object recognition plays the core role in the above applications. Therefore, this paper focuses on the foundational problem of 3D object recognition using MLS point clouds of road environments.

Traditionally, 3D object recognition was basically accomplished by designing hand-crafted descriptors. Some representative works include the Spin image (SI) (Johnson and Hebert, 1999), 3D shape context (3DSC) (Frome et al., 2004) and its variants (Tombari et al., 2010; Sukno et al., 2013; Dong et al., 2017), Fast Point Feature Histograms descriptor (FPFH) (Rusu et al., 2009), Signature of Histograms of OrienTations (SHOT) (Tombari et al., 2010), and Rotational Projection Statistics (RoPS) (Guo et al., 2013a). However, each of these methods catches only a part of the geometric characteristics of the 3D object. Therefore, the descriptiveness of existing 3D descriptors is still far from satisfactory.

To further enhance the discriminability of these traditional methods, it is natural to consider how to learn higher-level features to provide complementary information. One feasible way is to apply Deep Learning (DL) (Lecun et al., 2015), such as convolution neural networks (CNNs), to obtain deep features. Some research, using methods such as voxel-based (Wu et al., 2014; Maturana and Scherer, 2015), octree-based (Riegler et al., 2017; Wang et al., 2017), multi-view-based (Su

☆ Fully documented templates are available in the elsarticle package on CTAN.

* Corresponding author at: Fujian Key Laboratory of Sensing and Computing for Smart Cities, School of Information Science and Engineering, Xiamen University, China.

E-mail address: junli@xmu.edu.cn (J. Li).

https://doi.org/10.1016/j.isprsjprs.2019.01.024
Received 12 July 2018; Received in revised form 28 January 2019; Accepted 31 January 2019

et al., 2015) and point-set-based (Qi et al., 2016, 2017), has been undertaken to apply CNNs to 3D data analysis. Although the above methods have progressive improvement in the descriptiveness on some synthetic datasets (e.g. ModelNet40 (Wu et al., 2014)), the challenge remains to robustly process the MLS point clouds with outliers, noise and occlusion, which are all common and unavoidable in a real road environment.

In order to solve these problems, we explore the possibility of combining low-level features and the CNNs. However, applying CNNs that were designed for 2D images to analyze 3D point clouds is a nontrivial task. An image is organized with pixels that are represented as a fixed sequence and can be directly input to a CNN. However, 3D point clouds are unstructured and irregularly distributed. Therefore, they are unsuitable for CNN to process directly. To overcome this problem, we propose a 3D object recognition model, which builds a bridge between the 3D point clouds and CNNs. The key idea of our method is to represent the irregular 3D MLS point clouds as a series of regularly sampled 2D feature images and apply a designed fusion network to learn high-level features. To this end, we propose the following three feature descriptors generated for 3D data: Horizontal Single Spin Image (HSSI), Vertical Quantity Accumulation Image (VQAI), and Vertical Angle Accumulation Image (VAAI). These feature images can robustly preserve the local and global spatial relations of points. Moreover, these three descriptors all have the 2D form, so they can be directly fed into CNNs (e.g., AlexNet (Krizhevsky et al., 2012)). Then, a two-stage fusion network, which consists of a deep convolutional module and two fusion modules, is designed to fuse these multi-view features.

Our method was tested on a public urban road object dataset and two datasets acquired by a Mobile Laser Scanning (MLS) system. Experimental results fully exhibit that our method achieves superior performance in descriptiveness, robustness and efficiency. The main contributions of our work are as follows:

- Different from the traditional methods that catch only a piece of the geometric characteristics, we propose three feature descriptors (HSSI, VQAI and VAAI) to preserve the point distribution pattern of 3D shape from three different perspectives. These descriptors encode rich geometric information, such as the spatial relationship of neighboring points, which is still far from well employed in the existing point based methods.
- Using deep learning for 3D shape analysis is not straightforward. Therefore, by taking advantage of the 2D forms of three designed feature descriptors, which build a bridge between the 3D point clouds and deep learning, we propose a feature fusion framework to combine these feature descriptors with CNNs. In this way, high-level features can be learned to describe complex 3D shape.

The rest of this paper is organized as follows: Section 2 provides a literature review of 3D object recognition. Section 3 details the proposed method, including the computation of the three feature descriptors and the design of the two-stage fusion network. Section 4 presents the experimental results. Section 5 discusses both the experimental and comparative results. Section 6 concludes the paper.

2. Related work

In the last several decades, a number of solutions have been proposed for 3D object recognition. This section reviews some representative works. More complete overviews are presented in evaluation papers (Guo et al., 2014; Ioannidou et al., 2017). In general, existing 3D object recognition methods can be broadly categorized into two classes: Handcrafted-feature-based methods and Deep-learning-based methods.

Handcrafted-feature-based methods. Handcrafted-feature-based methods follow a traditional pipeline. Hand-crafted descriptors are designed to extract features from 3D objects, then those features are fed into off-the-shelf classifiers, such as SVMs. As one of the most popular local feature descriptors, Spin image (SI) (Johnson and Hebert, 1999) has been proved useful for object recognition, matching and modeling. However, the descriptiveness of SI is relatively limited. To improve the descriptiveness, Frome et al. (2004) proposed a 3D shape context (3DSC) descriptor by extending the 2D shape context method to 3D data. Several variants of 3DSC were also proposed, such as Unique Shape Context (USC) (Tombari et al., 2010), Asymmetry Patterns Shape Context (APSC) (Sukno et al., 2013) and 3D Binary Shape Context (BSC) (Dong et al., 2017). Besides, by accumulating the angle differences between a key point and its neighboring points, Rusu et al. (2009) proposed the Fast Point Feature Histograms descriptor (FPFH). In addition, Salti et al. (2014) generated the Signature of Histograms of OrienTations (SHOT) to encode local surface information. Guo et al. (2013a) and Guo et al. (2013b) introduced two local feature descriptors, Rotational Projection Statistics (RoPS) and Tri-Spin-Image (TriSI), respectively, both of which are descriptive and robust for many tasks. However, for these methods, the necessary step of triangulating the unstructured point clouds seriously impacts the time consumption and accuracy.

Deep-learning-based methods. Different from handcrafted-feature-based methods, deep-learning-based methods are end-to-end approaches, where the features and the classifiers are jointly learned from the data. However, because 3D unstructured point clouds are different from regular images, it is difficult to apply directly the CNNs to analyze 3D data. Therefore, the key problem of deep-learning-based methods is the design of representation of 3D data. Several works have been undertaken to handle this problem. They can be divided into three classes: volumetric methods, view-based methods and point-set-based methods.

Volumetric methods. It is straightforward to convert unstructured 3D point clouds to a regular grid data over which standard CNNs can be applied. Wu et al. (2014) proposed the ShapeNets by using the binary voxel grids as the representation of 3D data, while Maturana and Scherer (2015) generated the VoxNet by integrating the 3D volumetric occupancy grid. However, the above methods require high memory and computational cost as the voxel resolution increases. Recently, space partition methods, such as methods based on octree (Tatarchenko et al., 2017; Riegler et al., 2017; Wang et al.,2017) and k-d tree (Klokov and Lempitsky, 2017) were proposed to remedy the resolution issue. But these methods still rely on subdivision of a bounding volume rather than local geometric structure.

View-based methods. Multi-view method (Su et al., 2015) is the pioneering work that represents the 3D data with a set of images rendered from different views. These images are then fed as the input of CNNs to learn deep features. Several applications based on multi-view idea were also proposed. Chen et al. (2016) proposed a multi-view-based sensory-fusion framework to detect 3D objects for autonomous driving. Bai et al. (2016) developed a 3D shape retrieval system based on the projective images. Qin et al. (2018) fused the multi-view and multimodal representation of ALS point cloud for 3D terrain scene recognition. And Boulch et al. (2018) proposed the SnapNet by using the partial snapshots of 3D point clouds for the semantic segmentation. These multi-view methods encode the spatial relationship of points into 2D images and so can directly exploit the image-based CNNs for 3D shape analysis. However, it is unclear how to determine the number of views and how to distribute these views to cover the 3D shape while avoiding self-occlusions. Our method follows an idea similar to the multi-view CNNs. However, different from the way of generating 2D image in Su et al. (2015), we designed three feature descriptors, HSSI, VQAI, and VAAI, to obtain 2D representations of 3D point clouds, avoiding the view selection issue. More specifically, compared with multi-view method, the number of 2D feature images in our method is clear and small (3 global and 12 local images in our method). Additionally, because these feature images are obtained in different perspectives, the distribution of them would help to characterize the 3D shape.

Point-based methods. PointNet (Qi et al., 2016) is the pioneer in directly processing the point clouds. The advantages of PointNet lie in the following two factors. (1) It takes the point as the representation of 3D data, and (2) it uses the channel-wise max pooling, which is permutation invariant, to aggregate per-point features into a global descriptor. Therefore, PointNet not only avoids the issues arising from the generation of other representations of 3D data, such as time consumption, memory cost, and information loss, but also remains invariant to order permutation of input points. It has achieved impressive performance on 3D point cloud analysis. However, PointNet treats each point individually. In other words, it learns a mapping from 3D point clouds to the latent features without leveraging local geometric structure. This would not only make PointNet sensitive to noise, but also lose local geometric information among neighboring points. An improved architecture, PointNet+ + (Qi et al., 2017), exploits geometric features in local point sets and hierarchically aggregates these features for inference. However, PointNet+ + still treats individual points in local point sets independently and does not consider the relationship among point pairs.

In summary, existing handcrafted feature-based methods, while focusing on basic statistical information, give less consideration to the distribution patterns of points, which would lead to a deterioration in descriptiveness. DL-based methods learn deep features from 3D data to improve descriptiveness. However, most DL-based methods are applicable to synthetic datasets (e.g. ModelNet 40 (Wu20143D)), which are different from real-word scanning datasets. It is still a challenge for DL-based methods to robustly process 3D MLS point clouds with different noise. Compared to these methods (handcrafted feature-based and DL-based), this paper proposes three feature descriptors to robustly encode rich global and local shallow structure information, and develops a two-stage fusion network to learn high-level features to descriptively and robustly recognize 3D objects in real road environments.

3. The proposed method: JointNet

This section describes a generic recognition model and three feature descriptors. After that, these descriptors were combined with CNNs to design a two-stage fusion network, which consists of a deep convolutional module and two fusion modules. In other words, our framework, named JointNet, can connect the 3D feature descriptors and CNNs.

3.1. Generic recognition model

The recognition task can be formulated as a prediction problem that predicts the label given to one object. Therefore, the generic recognition model can be defined as a mapping F, which takes a set of 3D points $P = \{p_i \mid i = 1, \cdots, n\}$ as input and outputs k scores for all the k candidate categories. The mapping, F, is expressed as follows:

$$F\colon 2^{R^3} \to V,\ \ F(P) = v_{k\times 1}, \tag{1}$$

where R^3 denotes three-dimensional Euclidean space; $2^{R^3} = \{x \mid x \subset R^3\}$ is the power set of R^3; V stands for the vector space; P is a set of 3D points, $P \in 2^{R^3}$; $v_{k\times 1} \in V$ represents $k \times 1$ column vector; k is the number of classes; $F(P)$ measures the probability for all classes.

3.2. JointNet architecture

As the above analysis, the distribution pattern between points contains tremendous amount of structural information. The proposed method is expected to learn high level features from those information. Therefore, in JointNet architecture, three feature descriptors, named Horizontal Single Spin Image (HSSI), Vertical Quantity Accumulation Image (VQAI) and Vertical Angle Accumulation Image (VAAI), are designed to preserve the spatial relationship of the points. Feature maps generated by these descriptors can be considered as three kinds of representations of 3D point clouds. All these feature maps are in a 2D form and so suitable as the input of CNNs. Therefore, they build a bridge between 3D data and CNNs. Based on the generic recognition model and the above three feature descriptors, we proposed JointNet as follows:

$$\begin{aligned} F(P) &= g_2(g_1(f_1(P) + f_2(P) + f_3(P))) \\ &= g_2(g_1(f(T_1(P)) + f(T_2(P)) + f(T_3(P)))) \\ &= g_2(g_1(v_1 + v_2 + v_3)) = v, \end{aligned} \tag{2}$$

where

$$T_i\colon 2^{R^3} \to R^2,\ \ i = 1, 2, 3;\ \ T_1 = HSSI,\ \ T_2 = VQAI,\ \ T_3 = VAAI,$$

g_1 and g_2 represent the first and second fusion module, respectively. f represents the deep convolutional module, $v_i \in V$, $i = 1, 2, 3$, v_i represents $k \times 1$ column vector. For convenience, we denote $g_2(g_1(f(T_1)))$, $g_2(g_1(f(T_2)))$ and $g_2(g_1(f(T_3)))$ as *HSSINet*, *VQAINet* and *VAAINet*, respectively. Then the proposed model is a combined networks by fusing three sub-Nets.

As a supervised framework, JointNet consists of two stages: training and testing. Each stage has three steps: (1) computing three descriptors to capture shallow features, (2) feeding shallow features to a deep convolutional module to learn high-level features, (3) applying the fusion modules to generate the final prediction score-vectors for recognition. Fig. 1 shows the flowchart of JointNet. The following subsections introduce the three feature descriptors followed by the details of the feature extraction and fusion.

3.2.1. Horizontal Single Spin Image computation

The Spin image (SI), initially presented by Johnson and Hebert (1999), is a local descriptor that can be used in 3D object analysis, such as retrieval, recognition and registration. It has been considered to be the defacto benchmark for the evaluation of feature descriptors (Tombari et al., 2010; Guo et al., 2013a, 2014). To obtain the global spatial relationship among points, we design the first feature descriptor, HSSI, which follows an idea similar to SI. More specifically, the center of the object is chosen as the key point of HSSI and the vertical direction is chosen as the associative normal, n. This design is consistent with the fact that objects in road environment are often perpendicular to the road and rotate only around the vertical direction.

In our model, the ranges of X, Y and Z axis of 3D point cloud are normalized to $[-1,1]$, $[-1,1]$ and $[0,2]$, respectively. Therefore, the key point, p, of HSSI is the point, (0,0,1), and the normal, n, is chosen as (0,0,1). The projection function, S_{HSSI}, is described by the following formulas:

$$S_{HSSI}(p, q, \mathrm{n}) = (\alpha, \beta) = \left(\sqrt{(q_x)^2 + (q_y)^2},\ q_z\right) \tag{3}$$

where $p = (0,0,1)$; q is an arbitrary point in the 3D point clouds; q_x, q_y and q_z are the coordinates of q along the X, Y and Z axes, respectively. Then, the indexes (x, y) of q in the HSSI are defined by:

$$x = \left[\frac{\beta + 2}{2k_1}\right],\ \ y = \left[\frac{\alpha}{k_1}\right] \tag{4}$$

where $[\cdot]$ denotes the rounding function; $k_1 = \frac{2}{\sqrt{N_{HSSI}}}$; N_{HSSI} is the number of bins in HSSI. For convenience, we restrict the height and width of HSSI to be equal, and define $R_{HSSI} = \sqrt{N_{HSSI}}$ as the size of HSSI, where R_{HSSI} is an important parameter. Generally speaking, the larger the value of R_{HSSI}, the more information HSSI can encode, but sensitivity to noise also increases. According to Eq. (4), the indexes (x, y) of every point in HSSI can be calculated. So the number of points in every bin of HSSI can be counted. Thus, we obtain the global feature image. The generation of global image is shown in the upper right corner of Fig. 2(a). To obtain the local spatial distribution pattern, we divide each 3D object into four equal segments, as shown in the lower right corner of Fig. 2(a). Then, applying S_{HSSI} individually to these segments, we

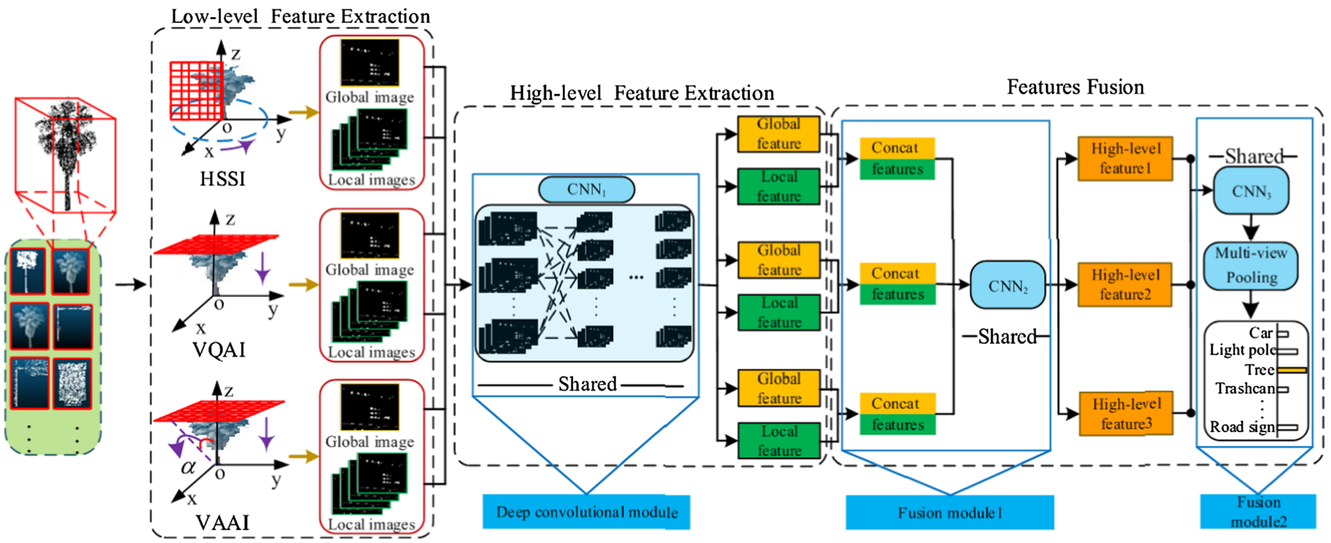


Fig. 1. Flowchart of our method including three parts: computing global and local feature maps using three descriptors; feeding feature maps to the deep convolutional module to learn high-level features; and applying two fusion modules to generate the final prediction score for recognition.

obtain four local feature images.

Fig. 2(b)–(e) show the same point clouds with increasing noise levels. Four corresponding global feature images generated by HSSI are depicted in Fig. 2(f)–(i). Obviously, the distribution patterns of pixels in four global feature images are similar. This indicates that HSSI can preserve effectively and robustly the spatial relationship among points.

3.2.2. Vertical quantity accumulation image computation

As the above discussion, HSSI captures the horizontal geometric structure. However, features along the vertical direction may also contain useful information. Therefore, considering only features along the horizontal direction tends to limit the descriptiveness. Thus, we design the second feature descriptor, VQAI, which obtains information along the distinctive vertical profiles.

It can be observed that objects under the road environments often have their own special vertical distribution patterns. For example, most of the points of a tree usually distribute around the trunk, while points on a light-pole often evenly distribute on the pole. Inspired by this observation, to capture the vertical distribution patterns, we propose a projection function, S_{VQAI}, by accumulating the number of points that lie in the same vertical position. It is defined as follows:

$$S_{VQAI}(p) \overset{def}{=} (p_x, p_y) \overset{def}{=} q, \tag{5}$$

where $p \in R^3$ is an arbitrary point in 3D data; p_x, p_y are the coordinates of p along the X, Y axes, respectively; $q \in R^2$ is the corresponding mapping point of p. Then the indexes (x, y) of q in VQAI are defined as:

$$x = \left[\frac{p_x + 1}{k_2}\right],\ y = \left[\frac{p_y + 1}{k_2}\right], \tag{6}$$

where $[\cdot]$ denotes the rounding function; $k_2 = \frac{2}{\sqrt{N_{VQAI}}}$; N_{VQAI} is the number of bins in the VQAI. For convenience, we restrict the height and width of VQAI to be equal, and define $R_{VQAI} = \sqrt{N_{VQAI}}$ as the size of VQAI, where R_{VQAI} is an important parameter. A larger value of R_{VQAI}, which might increase the ability to capture more meaningful information, may also weaken the robustness of VQAI. According to Eq. (6), the indexes (x, y) of every point in VQAI feature image are calculated. The number of points for every bin in VQAI feature image can be counted. Similarly, to obtain the local spatial distribution pattern, we divide a 3D object into four equal segments along the vertical direction, as shown in the lower right corner of Fig. 3(a). Then, by applying S_{VQAI} to these four segments, we obtain the local VQAI feature images.

The function mapping, S_{VQAI}, can be considered as compressing a 3D object into a 2D image in the vertical direction. Therefore, different from the horizontal information contained in HSSI feature images, VQAI feature images can preserve the vertical distribution pattern. Fig. 3(a) shows the generation of VQAI. Fig. 3(b)–(e) show the same point clouds with increasing noise levels. Four corresponding global VQAI feature images are depicted in Fig. 3(f)–(i). It can be observed

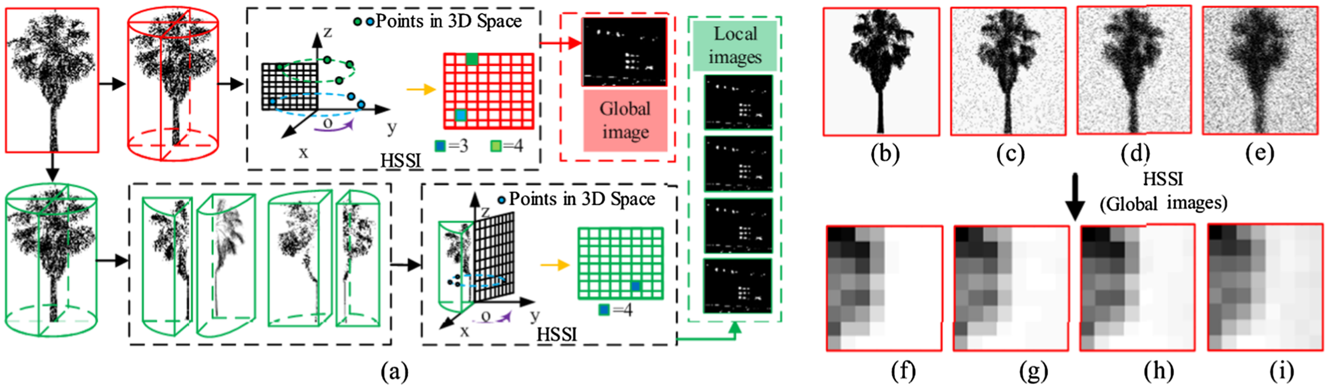


Fig. 2. An example of HSSI. (a) Generation of HSSI. (b) Original point clouds. (c) Point clouds with Gaussian noise with standard deviation of 0.01 and outliers ($\lambda = 0.1$), (d) Point clouds with Gaussian noise ($\delta = 0.03$) and outliers ($\lambda = 0.2$). (e) Point clouds with Gaussian noise ($\delta = 0.05$) and outliers ($\lambda = 0.3$). (f–i) Corresponding global feature images generated by HSSI for these point clouds shown in (b–e).

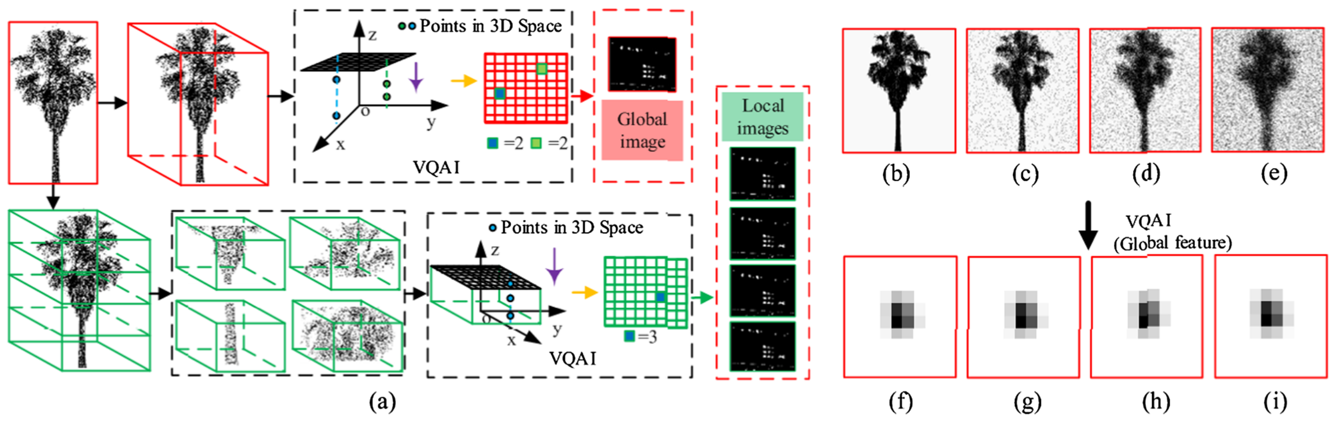


Fig. 3. An example of VQAI. (a) Generation of VQAI. (b) Original point clouds. (c) Point clouds with Gaussian noise with standard deviation of 0.01 and outliers ($\lambda = 0.1$), (d) Point clouds with Gaussian noise ($\delta = 0.03$) and outliers ($\lambda = 0.2$). (e) Point clouds with Gaussian noise ($\delta = 0.05$) and outliers ($\lambda = 0.3$). (f–i) Corresponding global feature images generated by VQAI for point clouds shown in (b–e), respectively.

that although there are different levels of disturbance in 3D shapes, the distribution patterns of the pixels in global feature images are similar. This result indicates that VQAI has strong robustness.

3.2.3. *Vertical angle accumulation image computation*

The descriptor, VQAI, which accumulates the number of points located in cells, contains rich information along the vertical direction. However, there may be some features that VQAI cannot capture. For example, as shown in the upper left of Fig. 3(a), the values of the blue and green cells in the global VQAI feature image are the same, but the distribution pattern is different. Actually, the two green points are close to each other, but the blue points are far away from each other.

Therefore, to improve the descriptiveness of VQAI, more features about the spatial information should be considered. To handle this problem, we first introduced the cosine measure. The cosine of the angle between vertical direction and the line connecting one point and the origin measures the spacial distance relationship among points located on the same vertical line. Then, for a point in 3D space, the key value K_p of the point is defined as the cosine of the above mentioned angle:

$$K_p \overset{def}{=} \cos((p_x, p_y, p_z + 1), (0, 0, 1)) \\ = \frac{p_z + 1}{(p_z + 1)^2 + (p_y)^2 + (p_z)^2}, \tag{7}$$

where p_x, p_y, p_z are the coordinates of p along the X, Y, Z axes, respectively. Building on the above observation and the cosine measure, we introduce the third feature descriptor, VAAI. It is formed by the projection function, S_{VAAI}, which is equivalent to the S_{VQAI}. So the indexes (x, y) of q in the VAAI feature image are defined by:

$$x = \left[\frac{p_x + 1}{k_3}\right],\ y = \left[\frac{p_y + 1}{k_3}\right] \tag{8}$$

where $[\cdot]$ denotes the rounding function; $k_3 = \frac{2}{N_{VAAI}}$; N_{VAAI} is the number of bins in VAAI. The $R_{VAAI} = \sqrt{N_{VAAI}}$ is defined as the size of VAAI. Different from VQAI, in VAAI, the value of a cell rather than the number of points in this cell, is calculated as follows:

$$w_{i,j} = \sum_{l=1}^{n} K_{p_l}, \tag{9}$$

where $w_{i,j}$ is the value of cell with index (i, j); p_l is the point located on the cell with index (i, j); K_{pl} is the key value of p_l, and the key value is defined in Eq. (7). Thus, we obtain the global VAAI feature image. To obtain the local spatial distribution pattern, we divide a 3D object into four equal segments along the vertical direction, as shown in the lower right corner of Fig. 4(a). Then, by applying S_{VAAI} to these segments, we obtain the local VAAI feature images. Fig. 4(a) shows the generation of VAAI. Fig. 4(b)–(e) show the same point clouds with increasing noise levels. Four corresponding global VAAI feature images are shown in Fig. 4(f)–(i). We can find that various disturbance results in similar global feature images. This result shows that VAAI is robust to noise.

3.2.4. *High-level feature learning via CNNs*

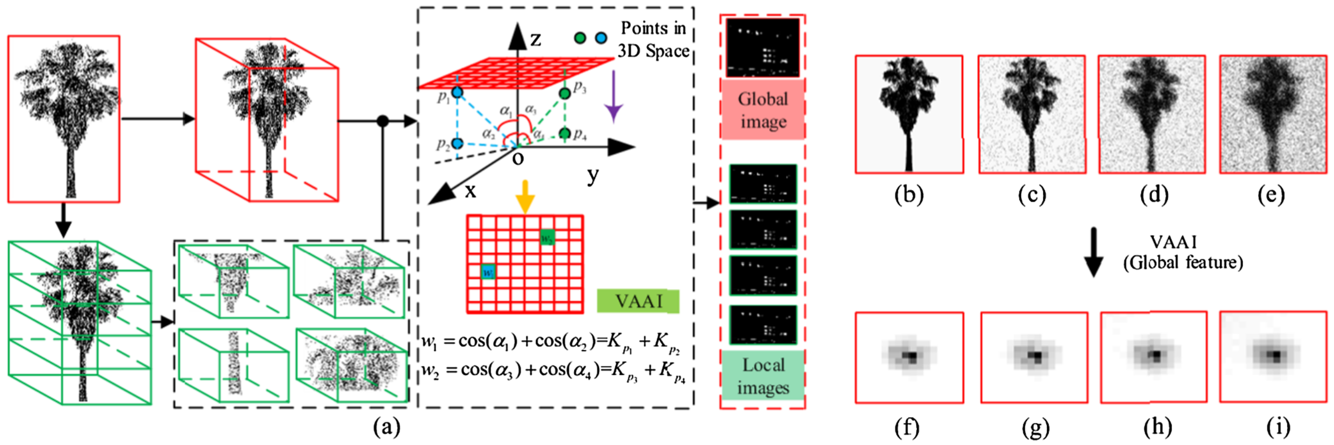
The above feature images describe the basic low-level statistical features. To obtain more complementary information, it is feasible to further extract high-level features from the basic statistical results. Recently, deep CNNs have achieved huge success in many image processing tasks, such as image classification, object detection, semantic segmentation (Girshick, 2015; Ren et al., 2015; He et al., 2017). Therefore, we adopt CNNs as our high-level feature learning module. However, due to the intrinsic differences between the structures of 3D point clouds and 2D images, it is not a simple task to input directly the raw point clouds to CNNs. Fortunately, feature images obtained by HSSI, VAAI and VQAI, are in 2D form, which is suitable for the input of CNNs.

Overview of JointNet. As shown in Fig. 1, the architecture of JointNet consists of three parts: computing three descriptors to capture low-level features; feeding feature images to deep convolutional module to learn high-level features; and generating the final prediction score-vector using the two fusion modules, to recognize objects.

Deep convolutional module. The deep convolutional module is used to extract high-level features. The multi-view feature images, generated by these three descriptors, are used as the bridge between deep convolutional module and 3D data. As shown in left part of Fig. 5, deep convolutional module mainly contains five convolutional (Conv) layers, a spatial pyramid pooling (SPP) layer (He et al., 2015), two fully connected (FC) layers and a reshape layer. Max pooling and 2-level pyramid are used as the parameters of the SPP layer.

Fusion module1. As shown in the middle of Fig. 5, fusion module1 consists of a connected operation and two Conv layers. The connected operation concatenates the global and local feature images to obtain a multi-channel feature image. The two Conv layers learn higher level features from the multi-channel feature image.

Fusion module2. As shown in the right of Fig. 5, fusion module2 contains two parts: the CNN3 and multi-view pooling. CNN3 consists of four Conv layers and two FC layers. The multi-view pooling is placed after the last FC layer. Inspired by the TL-pooling operator (Laptev et al., 2016) and the view-pooling layer (Su et al., 2015), the multi-view pooling is achieved by using element-wise summation. It computes the summation of score vectors at the same index i:



$w_1 = \cos(\alpha_1) + \cos(\alpha_2) = K_{p_1} + K_{p_2}$
$w_2 = \cos(\alpha_3) + \cos(\alpha_4) = K_{p_3} + K_{p_4}$

Fig. 4. An example of VAAI. (a) Generation of VAAI. (b) Original point clouds. (c) Point clouds with Gaussian noise with standard deviation of 0.01 and outliers ($\lambda = 0.1$), (d) Point clouds with Gaussian noise ($\delta = 0.03$) and outliers ($\lambda = 0.2$). (e) Point clouds with Gaussian noise ($\delta = 0.05$) and outliers ($\lambda = 0.3$). (f–i) Corresponding global feature images generated by VAAI for these point clouds shown in (b–e).

$$S_i = \sum_{h=1}^{n} V_i^h, \tag{10}$$

where S is the final prediction score vector, $i = 1, 2, \ldots, k$, k is the number of categories, n is the number of CNN3 branches, and V^h is the score vectors output from h-th CNN3 branch.

In addition, the loss function is taken as the classic softmax cross entropy loss:

$$L_{Joint\,Net} = E(t, y) = -\sum_{i=1}^{N} t_i \log S_i, \tag{11}$$

where t_i, which takes the form of one-hot representation, is the label of the ith training sample; N is the batch size; S_i is the final prediction score vector for ith training sample. Then we have the model objective as:

$$G = arg\ \min L_{Joint\,Net} \tag{12}$$

Thus, after training the model, the class label can be obtained from the prediction score vector.

Training details. Our method was implemented with Tensorflow on a NVIDIA GTX1080Ti.

We optimized the networks using stochastic gradient descent (SGD) which is provided in Tensorflow. Note that, there are several parameters needed to be optimized, e.g., the batch size, the initial learning rate, the momentum of SGD, and dropout rate. The grid search method was used to find the optimal combination. The batch size, initial learning rate, the momentum of SGD, and dropout rate were set with ranges of (16, 32, 64), (0.01, 0.001, 0.0001), (0.7, 0.8, 0.9) and (0.6, 0.7, 0.8), respectively. The accuracy (defined in Eq. (17)) was used as the metric. After testing the performance of JointNet with all possible combinations, we obtained the optimal setting: {32, 0.01, 0.9, 0.7}. Therefore, we trained our model using SGD with a momentum of 0.9, a weight decay of 0.0005 and batch size of 32. The initial learning rate was set to 0.001, which decreases by half in every 20 epochs. 150 epochs were used for training step.

4. Experiments

Several experiments were conducted to evaluate the proposed method. Section 4.1 introduces three datasets used in this work. Section 4.2 generates the optimal parameters. Section 4.3 analyzes the contributions of HSSINet, VQAINet and VAAINet to JointNet. Section 4.4, 4.5 and 4.6 evaluate the descriptiveness, robustness and efficiency of JointNet, respectively.

4.1. Dataset description

We evaluated the performance of JointNet on three datasets, i.e., HDRObject9, SMDObject6 and Sydney Urban Objects (Deuge et al., 2013).

HDRObject9 and SMDObject6. We collected experimental data sets, HDRObject9 and SMDObject6, by our MLS system on Huandao Road and Siming District, Xiamen, respectively. Our equipment, a MLS system, integrates two full-view RIEGL VQ-450 laser scanners that generate a maximal effective measurement rate of 1.1 million measurements per second and a line scan speed of up to 400 scans per second (see Fig. 6(a)). The accuracy and precision of the scanned point clouds are within 8 mm and 5 mm, respectively. There are three steps to acquire our datasets.

Firstly, points that are far from the trajectory and non-object points lying in the air were all removed to make sure the raw data is clean. Secondly, we extracted and removed the ground and building facade

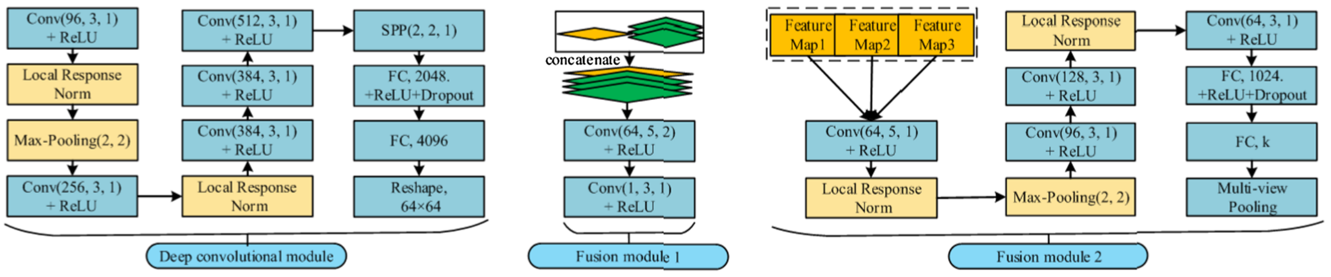


Fig. 5. Networks of JointNet. Left: network of convolutional module; Middle: network of the first fusion module; Right: network of the second fusion module. Conv (*m, k, s*) represents m convolution kernels of size k × k with a stride of *s* steps. Max Pooling (*k, s*) stands for the k × k max pooling operation with a stride of s steps. SPP (2, 2, 1) means 2-level pyramid are used in the SPP layer. The sizes of the first and second pyramids are 2 × 1 and 1 × 1, respectively.

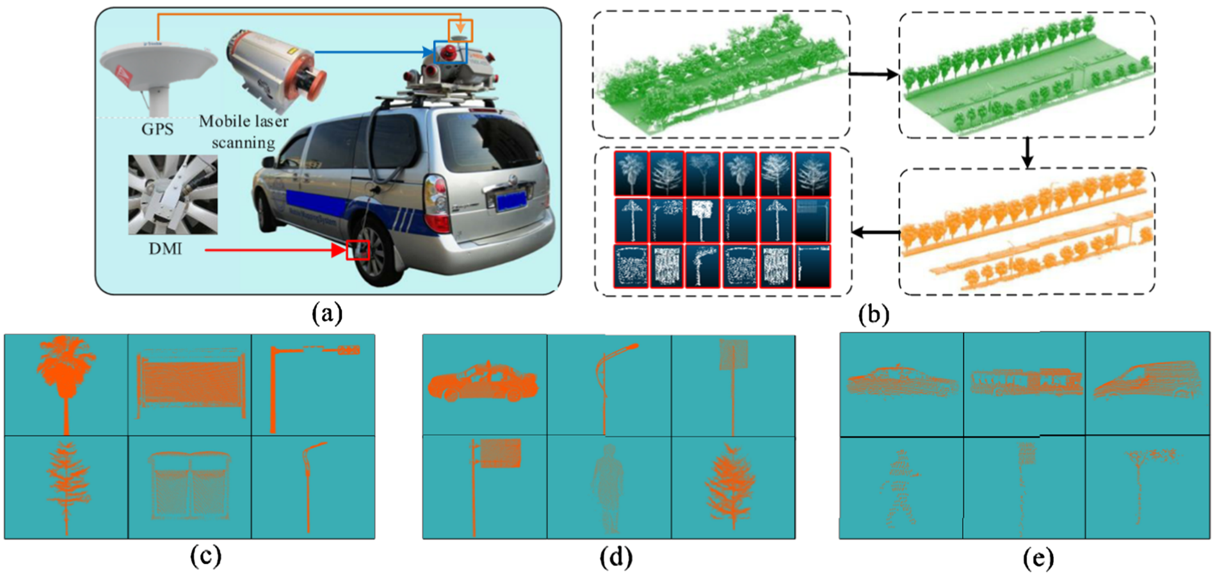


Fig. 6. (a) MLS system. (b) Flowchart of data preprocessing. (c)–(e) Samples from HDRObject9, SMDObject6 and Sydney Urban Objects, respectively.

points in order to separate the remaining objects from each other. Specifically, to remove ground points from scene, we used the voxel-based upward growing approach (Yu et al., 2015b), which achieves good performance both on accuracy and efficiency for preprocessing MLS point clouds. To remove the building facade points, the voxel-based normalized cut segmentation method proposed in Yu et al. (2015c) was applied to the off-ground points to partition them into separated clusters. Then, considering the fact that the building facade points are usually located farther away from the road than the other objects, such as the road signs, trees and bus stations, we removed those clusters that are away from the track line with a given threshold of 10 m. Note that, the possible disturbances in this step, such as noise, occlusion and outliers, may have influence on the performance of the proposed method. Section 4.5 will evaluate the effect of these possible disturbances. Thirdly, the remaining points were segmented into objects with their labels by manual method. Fig. 6(b) shows the preprocessing flowchart.

In HDRObject9, there are 9 object classes (the number of instances in each class is indicated in parenthesis): Bus station (1143), Light-pole (4175), Road sign (1281), Station sign (2062), Traffic light(1728), Traffic sign(4687), Trashcan (1456), Tree1 (Plam) (4200) and Tree2 (Cycas) (1920). In SMDObject6, there are 6 object classes: Car (1265), Light-pole (2500), Pedestrian (7 4 0), Road sign (8 5 5), Traffic sign (2930), Tree3 (Ficus microcarpa) (1203). Fig. 6(c) and (d) show samples from HDRObject9 and SMDObject6, respectively. Each dataset was split into 70%, 10%, and 20% subsets for training, validating and testing, respectively.

Sydney Urban Objects (SUObject14). This dataset was generated from several sequences of Velodyne scans by applying the segmentation techniques developed in Deuge et al. (2013). It contains 588 labelled objects in 14 categories (vehicles, pedestrians, signs, and trees) and is divided into four folds. Unlike HDRObject9 or SMDObject6, this dataset is formed by very sparse point clouds. It demonstrates non-ideal sensing conditions with occlusions (holes) and a large variability in viewpoint. Hence, it would be a challenging task to recognize objects from this dataset. Fig. 6(e) shows several samples.

4.2. JointNet generation parameters

JointNet has three important parameters: the sizes of HSSI, VQWI and VAAI. To obtain the appropriate sizes, JointNet was tested under different parameter settings using the Precision-Recall (PR) curve on HDRObject9 with different disturbances. We disturbed the HDRObject9 by down-sampling, adding Gaussian noise and outliers. (η, δ, λ) is denoted as the combination of the noise parameters, where η, δ and λ represent the down-sampling ratio, the standard deviation of Gaussian noise, and the rate of outliers, respectively. Because the ranges of X, Y and Z axis of 3D point clouds are normalized to $[-1,1]$, $[-1,1]$ and $[0,2]$, respectively, the δ has no unit. The rate of outliers is calculated as follows:

$$\lambda = \frac{number\ \ of\ \ outliers}{(number\ \ of\ \ outliers) + (number\ \ of\ \ originpoints)} \tag{13}$$

Then HDRObject9 with (0.5, 0.01, 0.1), (0.5, 0.03, 0.2) and (0.5, 0.05, 0.3) are denoted as test dataset1, dataset2 and dataset3, respectively.

PR curve generation. First, after feeding the low-level feature images generated by HSSI, VQAI and VAAI, into JointNet, we obtain the final score-vector, $\boldsymbol{S}$, for each testing sample. Second, for the ith category, if the distance $||1 - \boldsymbol{S}_i||$ is less than a threshold, τ, the corresponding testing sample is predicted as a sample of the ith category. Then, if this sample belongs to the ith category, it is a true positive sample of the ith category; otherwise it is a false positive sample. Thus, the precision and recall of the ith category can be calculated as follows:

$$\Pr ecision = \frac{TP}{TP + FP} \times 100 \tag{14}$$

$$\operatorname{Re} call = \frac{TP}{N_i} \times 100 \tag{15}$$

where TP is the number of true positive samples, FP is the number of false positive samples, and N_i is the number of samples in the ith category. Finally, the average precision and recall for all categories can be calculated. By varying the threshold, τ, the PR curve is generated. Ideally, the PR curve will fall in the top-right corner of the plot, which means the method obtains both high precision and recall (Davis and Goadrich, 2006; Guo et al., 2016).

Note that, we evaluated the performance of JointNet with examining 6 (options for R_{HSSI}) × 6 (options for R_{VQAI}) × 6 (options for R_{VAAI}) = 216 combinations, and found the optimal combination {16,8,16}. Considering that the number of combinations is so large, we presented the performance of each parameter using the controlling variable method. More specifically, we only changed one parameter, while fixed the other two parameters to their optimal values.

Size of *HSSI*. The R_{HSSI} plays an important role in JointNet. A larger

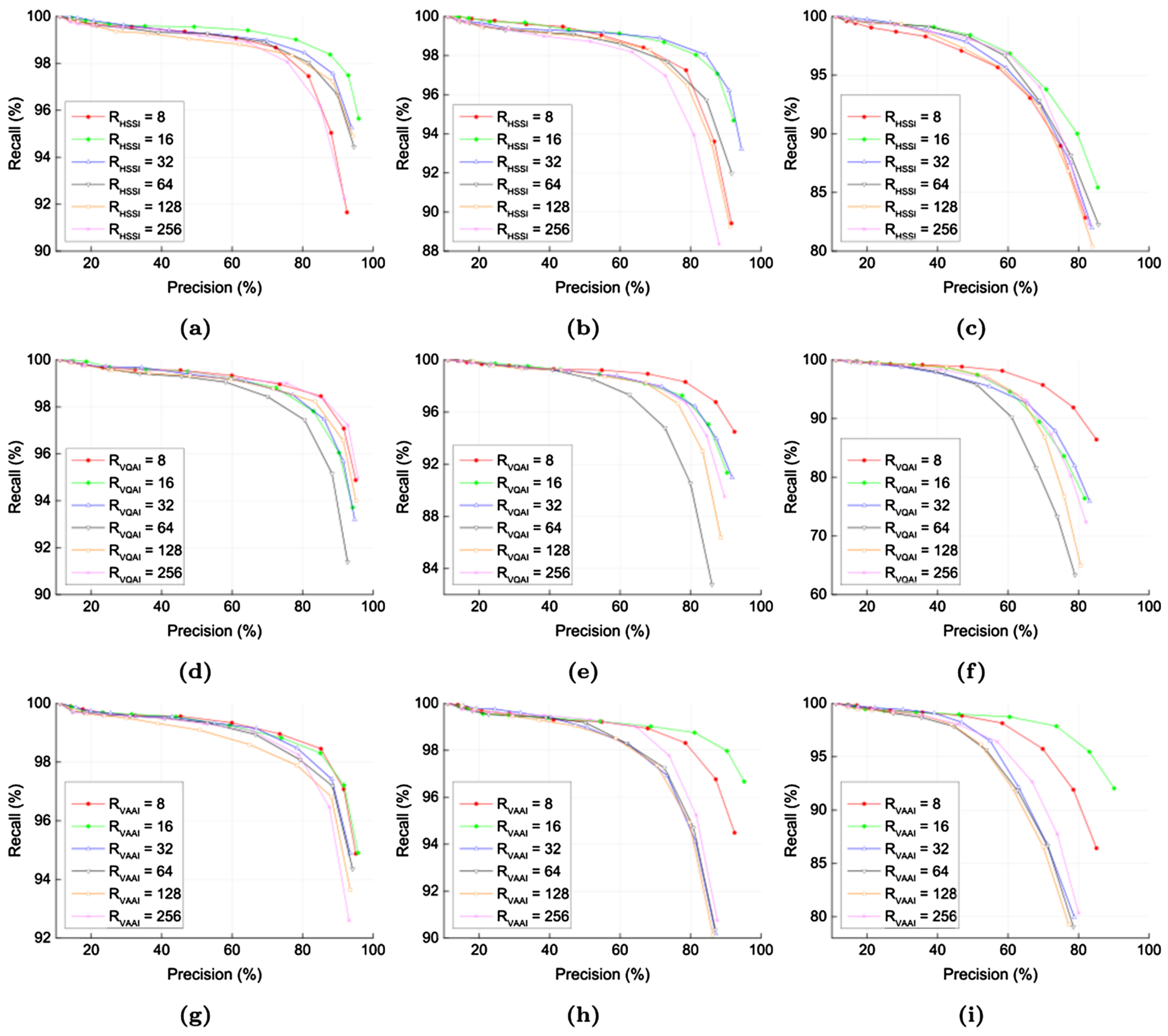


Fig. 7. PR curves with different parameter combinations. (a–c) Varying R_{HSSI} on test data set1, set2 and set3, respectively. (d-f) Varying R_{VQAI} on test data set1, set2 and set3, respectively. (g–i) Varying R_{VAAI} on test data set1, set2 and set3, respectively. (We set R_{HSSI} = 16, R_{VQAI} = 8 and R_{VAAI} = 16 as the optimal parameters to balance the descriptiveness and robustness).

value of R_{HSSI} enables HSSINet to capture more information from point clouds. However, HSSINet also becomes more sensitive to noise. To obtain the optimal value, we tested the performance of JointNet by varying R_{HSSI} on three test datasets, while setting the other two parameters, R_{VQAI} and R_{VAAI} as 8 and 16, respectively. Fig. 7(a)–(c) show the PR curves on three test datasets with values of R_{HSSI} ranging from 8 to 256. As shown in Fig. 7(a), the performance of JointNet improves as R_{HSSI} increases from 8 to 16. This is because *HSSI* captures more details as the R_{HSSI} increases. However, the performance of JointNet deteriorates as R_{HSSI} increases. The reason is that a large value of R_{HSSI} increases the sensitivity to noise. Fig. 7(c) shows a similar situation. However, as shown Fig. 7(b), the performance of JointNet with R_{HSSI} = 32 is somewhat better than that with R_{HSSI} = 16. To obtain the proper balance between capturing ability and maintaining high robustness, we set R_{HSSI} = 16 as the optimal value.

Size of *VQAI*. R_{VQAI}, which determines both the descriptiveness and robustness of the VQAINet, is another important parameter. We tested the performance of JointNet by varying R_{VQAI} on three test datasets, while maintaining R_{HSSI} and R_{VAAI} at 16 each. Fig. 7(d)–(f) illustrate the PR curves for different R_{VQAI}. As shown in Fig. 7(d), JointNets, with R_{VQAI} = 8 and R_{VQAI} = 256, achieve excellent performance on test dataset1. However, as shown in Fig. 7(e) and (f), JointNet with R_{VQAI} = 8 achieves the best performance, while the performance of JointNet with R_{VQAI} = 256 deteriorates sharply. This is because a smaller value of R_{VQAI} is more robust to noise. Therefore, we set R_{VQAI} = 8.

Size of *VAAI*. We evaluated the performance of JointNet by varying R_{VAAI}, while maintaining R_{HSSI} and R_{VQAI}, at 16 and 8, respectively. Fig. 7(g)–(i) show the PR curves for different R_{VAAI}. As shown in Fig. 7(g), JointNets with R_{VAAI} = 8 and R_{VAAI} = 16 achieve excellent performance on test dataset1. As shown in Fig. 7(h) and (i), the performance of JointNet improves as R_{VAAI} increases from 8 to 16. That is, as the R_{VAAI} increases, more information is encoded, resulting in the significant improvement. However, as R_{VAAI} continues increasing, the performance degrades sharply. This is because increasing R_{VAAI} not only adds very little significant information to the model, but also increases the sensitivity to noise. Thus, the optimal value can be set as R_{VAAI} = 16.

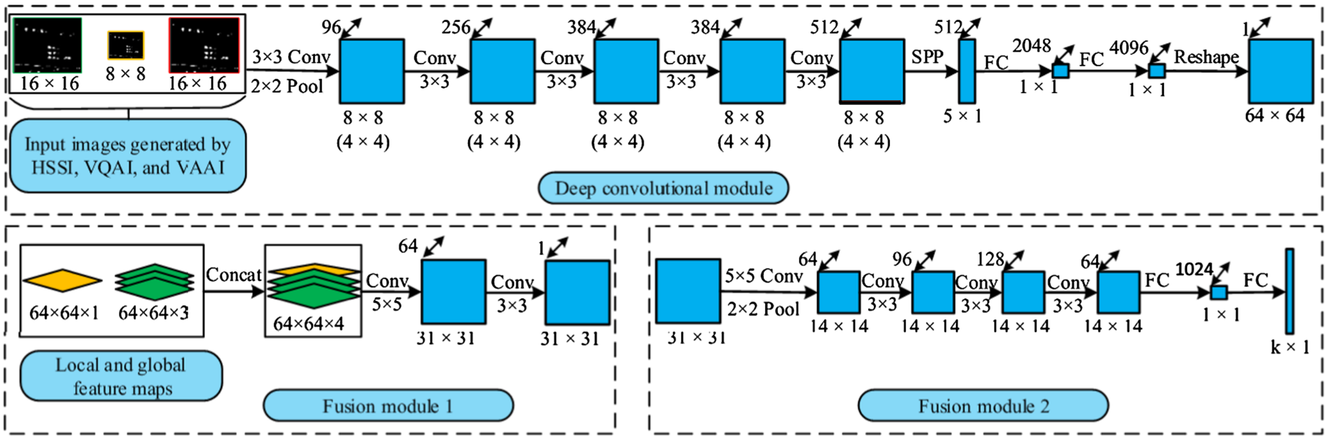


Fig. 8. Network parameters and the sizes of feature images in the three modules.

4.3. Contributions of three sub-Nets

As discussed in Section 3, JointNet consists of three sub-Nets: HSSINet, VQAINet and VAAINet. HSSINet captures rich information of the 3D object along the horizontal direction, while VQAINet and VAAINet describe the distribution patterns of points along the vertical direction from the perspectives of quantity and angle, respectively. This subsection evaluates the contributions of three sub-Nets to JointNet. F1 score and confusion matrix are used as the evaluation criteria. The F1 score is computed as follows:

$$F1 = 2 \times \frac{\Pr ecision \times \operatorname{Re} call}{\Pr ecision + \operatorname{Re} call} \times 100 \tag{16}$$

where precision and recall follow Eqs. (14) and (15).

Fig. 8 shows the network parameters in three modules. It illustrates the size of convolution kernel, the number of outputs in each convolution layer and also the sizes of the generated feature images. Note that, the network architecture of each sub-Net is similar to the deep convolution module. The difference is that a fully connected layer is used to replaced the reshape layer.

Fig. 9(a)–(c) show F1 scores on test dataset1 to dataset3, respectively. On one hand, JointNet achieves a higher F1 score for nearly every category than three sub-Nets. Especially, as shown in Fig. 9(a), JointNet has a excellent F1 score (about 90%) for each category, while F1 scores generated by most of sub-Nets are less than 90%. This indicates that JointNet has a better descriptiveness than sub-Nets. On the other hand, HSSINet performs well on road signs, station signs, Plam and Cycas, while VQAINet provides a competitive level of F1 score on road signs and traffic signs, and VAAINet achieves a higher F1 score than the other two sub-Nets on bus stations and traffic lights. Note that, road signs, station signs, Plam and Cycas contain more horizontal features; while samples belonging to bus stations or traffic lights have more discriminating vertical information. Thus, these results are consistent with the functions of three sub-Nets: HSSINet captures more horizontal features from 3D shape, while VQAINet and VAAINet encode more vertical information. In summary, these three sub-Nets make their contributions to JointNet from different perspectives.

Fig. 10 shows the confusion matrices computed by three sub-Nets and JointNet. Fig. 10(a)–(c) and (d) show the results generated by three sub-Nets and JointNet on test dataset1, respectively. It is seen that after fusing three sub-Nets, the values of the diagonal elements in the confusion matrix increase, while the values of some non-diagonal elements decrease. This means that the intra-class similarity increases, while the inter-class similarity decreases. As shown in the second and third rows of Fig. 10, JointNet also achieves the best performance over three sub-Nets on test dataset2 and dataset3, respectively. In addition, from the first to third columns of Fig. 10, it can be observed that HSSINet achieves better performance than the other two sub-Nets. This demonstrates that HSSINet has a higher descriptiveness and so it may make more contribution to JointNet. Consequently, by fusing these three sub-Nets, JointNet improves discrimination.

Although fusing sub-Nets would improve the descriptiveness, it may affect the space (model size) and time complexity (forward and backward operation) of JointNet. To evaluate the influence, we ran 1000 forward-backward iterations for each sub-Nets and JointNet, and then calculate the average time. We also recorded the peak GPU memory consumption and the time of recognizing testing samples on both HDRObject9 and SMDObject6. Table 1 shows the results. Obviously, after fusing sub-Nets, model size and GPU memory consumption increase by more than 60 MB and 1000 MB, respectively. Besides, the

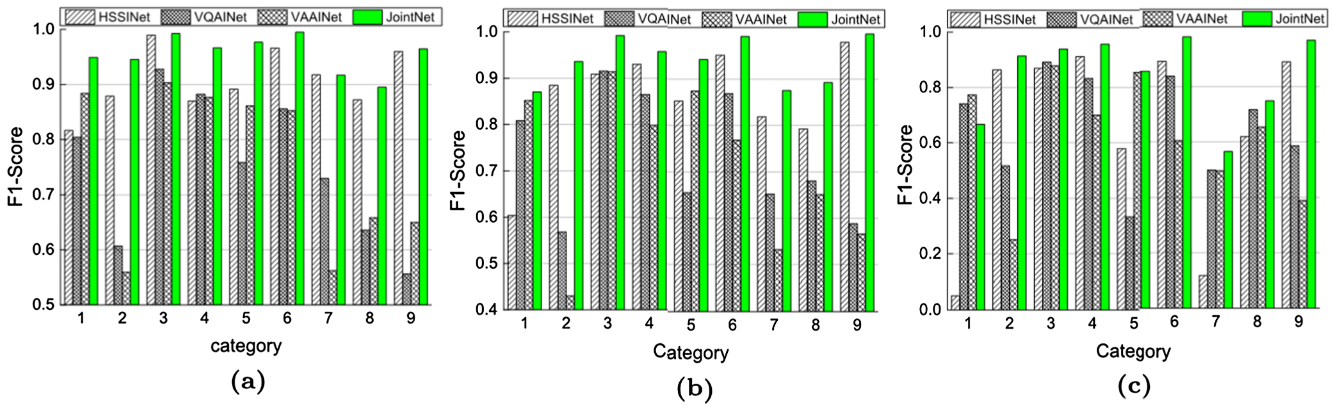


Fig. 9. Comparison of F1 scores generated by sub-Nets and JointNet on three test datasets. (a)–(c) test dataset1, dataset2 and dataset3, respectively.

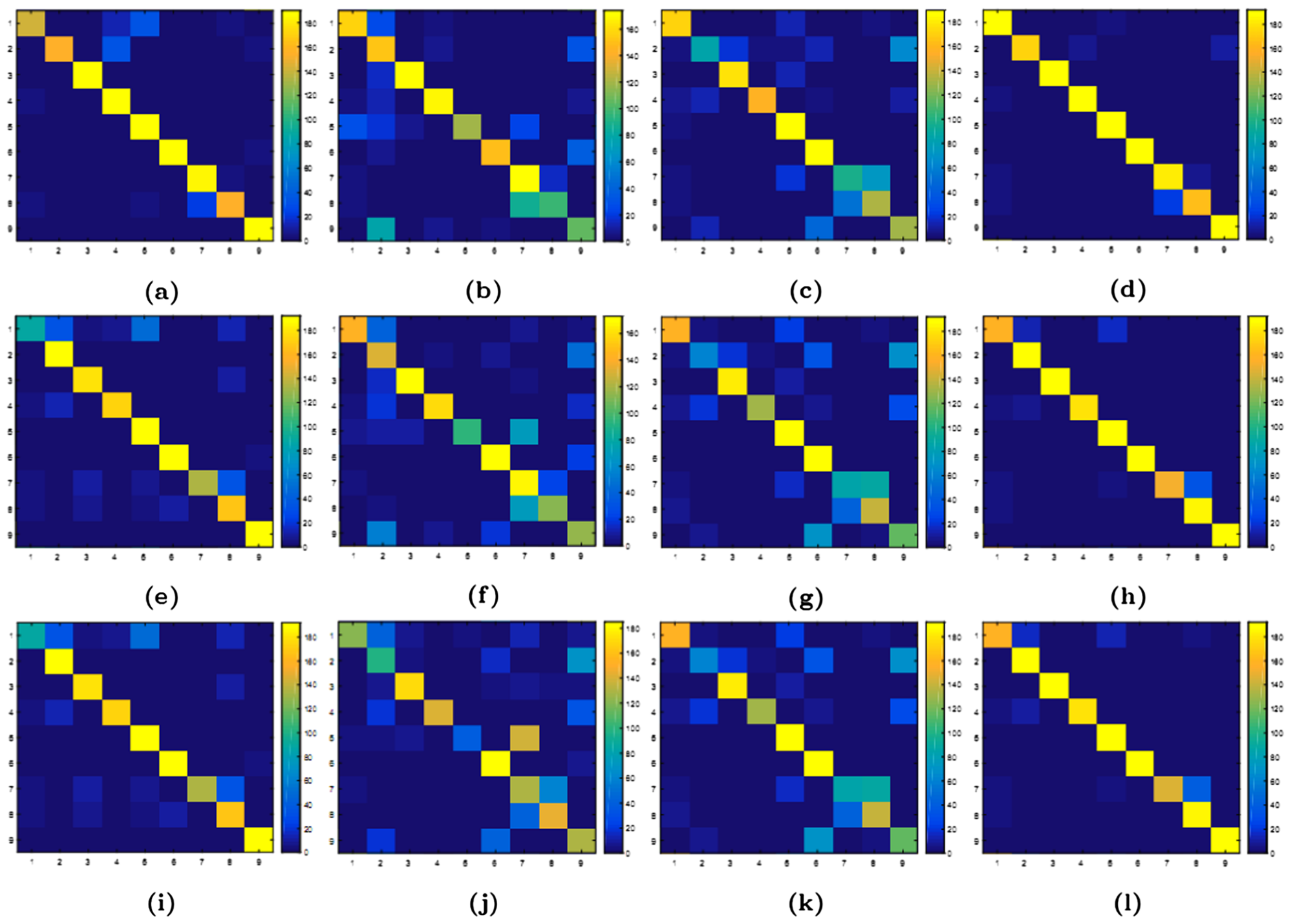


Fig. 10. Confusion matrixes generated by sub-Nets and JointNet on three test datasets. (a)–(d) test dataset1, (e)–(h) test dataset2, (i)–(l) test dataset3, respectively.

time of one forward-backward operation in JointNet is about three times than that in each sub-Net. The time of recognizing testing samples on each dataset increases about 1.5 times after fusing three sub-Nets. These results indicate that fusing sub-Nets would reduce the time and memory efficiency. The reason lies in that fusing sub-Nets increases the number of layers and parameters of the network.

4.4. Descriptiveness comparison

To demonstrate the descriptiveness of the proposed method, we compared JointNet with four competing methods: Sping Image (SI) (Johnson and Hebert, 1999), Signature of Histograms of OrienTations (SHOT) (Tombari et al., 2010), Hierarchy Descriptor (HD) (Bertrand, 2009) and PointNet (Qi et al., 2016). SI is the most popular 3D local feature descriptor that has achieved good performance. SHOT obtains a good balance between descriptiveness and robustness. HD is a context-based model, which describes the typical vertical features of 3D shape observed in urban environments. PointNet is a point-based model that exhibits superior feature extraction performance on some synthetic datasets. PR curve and F1 score were used to measure the performance.

Fig. 11 shows PR curves generated by selected methods on three datasets. JointNet (red curve) achieves superior performance on HDRObject9 and SMDObject6 (Fig. 11(a) and (b)). The recall of JointNet remains above 95% and 90% on these two datasets, respectively. Besides, as shown in Fig. 11(c), JointNet obtains a set of competitive recall values on Sydney Urban Objects dataset. When precision exceeds 30%, the recall of JointNet is larger than that of PointNet. In addition, Table 2 shows the numerical values on three datasets. (In this experiment, for Sydney Urban Objects dataset, the fold 4 is selected as testing data and other three folds are training data.) Obviously, JointNet achieves higher precision, recall, and F1 score than other methods on each datasets. Therefore, our method demonstrates promising discriminative capacity for 3D point clouds.

Since the Sydney Urban Objects dataset is a public dataset, results of several state-of-the-art algorithms are available. Therefore, using the same testing protocol, we compared our method with these existing methods. Different from the protocol used in the above experiment, we evaluated the F1 score achieved on each fold and obtained the average value. Table 3 shows the results. It is clear that DL-based methods outperform the handcrafted-feature-based methods, such as

Table 1
Space and time complexity of sub-Nets and JointNet. The model size and GPU-memory consumption are used to evaluate the space complexity. The time of forward-backward operation and the time of recognition on two data sets are used to evaluate the efficiency.

Method	Size of model	GPU-memory consumption	Time of one backward operation	Time of one forward operation	Time of recognition on HDRObject9	Time of recognition on SMDObject6
Sub-Net	159 MB	14264 MB	38.06 ms	11.53 ms	17.16 s	4.54 s
JointNet	223 MB	15647 MB	109.13 ms	33.91 ms	29.21 s	7.69 s

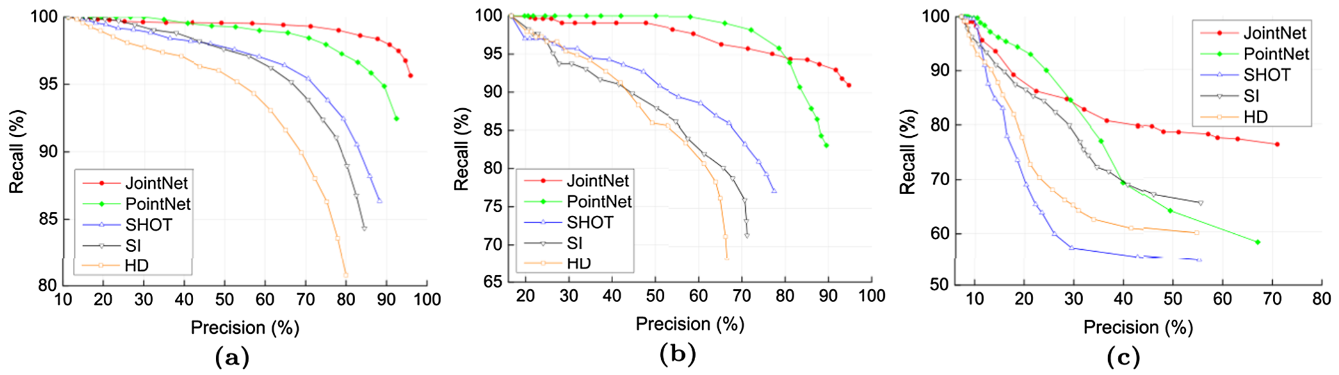


Fig. 11. PR curves generated by four selected methods and JointNet on three datasets. (a)–(c) HDRObject9, SMDObject6 and Sydney Urban Objects, respectively.

unsupervised feature learning (UFL) (Deuge et al., 2013) and Global Fourier Histogram descriptor (Chen et al., 2014). Our method surpasses the VoxNet (Maturana and Scherer, 2015), which is the pioneer work applying CNNs to 3D data. Additionally, compared to the BV-CNNs (Ma et al., 2018), our method achieves a competitive result (0.755 for BV-CNNs and 0.749 for JointNet). However, other DL-methods, including the ORION (Sedaghat et al., 2017) and LightNet (Zhi et al., 2017), obtain higher scores than JointNet. Specially, the LightNet achieves the best performance among all selected methods. The reason may be that LightNet has a very small number of network parameters and so it is hard to overfit, which is the key issue for recognition on dataset with large viewpoint variations and occlusion, such as the Sydney Urban Objects dataset. This result inspires us to reduce the model size of JointNet in our future work.

4.5. Robustness comparison

A method proposed to process 3D point clouds is robust if it is insensitive to many disturbances, which can be generated by noise, outliers, occlusion, or variations in the density of point clouds (Tombari et al., 2013). By performing experiments with a set of disturbances, including Gaussian noise, down sampling, adding outliers, and occlusion, we evaluated the robustness of our method and other four selected methods. Accuracy, used to measure the performance, is calculated as follows:

$$Accuracy = \frac{\sum_{i=1}^{k} TP_i}{\sum_{i=1}^{k} N_i} \times 100 \tag{17}$$

where TP_i is the number of true positive samples in the ith category; N_i is the number of samples in the ith category. k is the number of categories.

Firstly, we added different Gaussian disturbance on three datasets. Specifically, for a given standard deviation, Gaussian noise with zero mean was added to the x, y and z axis of every point. Fig. 12(a),(e) and (i) show the accuracy curves generated by selected methods on HDRObject9, SMDObject6 and Sydney Urban Objects, respectively. It is seen that our method has a significant advantage over other four selected methods. Especially, as shown in Fig. 12(a), even when the standard deviation is 0.1, the accuracy of our method is still 80%. In addition, PointNet is sensitive to Gaussian noise. Although held at a high level when the standard deviation is less than 0.04, the accuracy of PointNet deteriorates sharply as the standard deviation increases.

Secondly, we evaluated the robustness of the selected methods to varying point density. Testing samples on three datasets are downsampled with different rates. Fig. 12(b), (f) and (j) show the results on HDRObject9, SMDObject6 and Sydney Urban Objects, respectively. Obviously, JointNet achieves the best performance over all the tested methods. Especially, as shown in Fig. 12(b), the accuracy of JointNet is higher than 80% even when the rate is 1/64. This indicates that our method is robust to varying point density.

Thirdly, we added different rates of outliers to three datasets. Fig. 12(c), (g) and (k) show the compared results. As shown in these three sub-figures, our method yields the best performance among all methods. Although the testing data is disturbed by large rate of outliers, JointNet still achieves high discrimination. Especially, as shown in Fig. 12(c), the accuracy of JointNet is still larger than 90%, even the rate of outliers is 30%. We can also observe that PointNet is very sensitive to outliers, while SHOT has a higher robustness on HDRObject9.

Finally, we analyzed the performance of the selected methods with respect to occlusion. The definition of occlusion for 3D data in Johnson and Hebert (1999), which is based on the mesh, is not suitable for point clouds. Therefore, considering that all points are normalized to fixed size space (x: [−1,1], y: [−1,1], z: [0,2]), we define the occlusion by deleting points located in occluded space with given size. The occlusion rate is defined as follows:

Table 3
F1 score achieved by different methods on the Sydney Urban Objects dataset.

Method	Average F1 score (%)
GFH + SVM (De Deuge et al., 2013)	0.67
GFH + SVM (Chen et al., 2014)	0.71
VoxNet (Maturana and Scherer, 2015)	0.72
BV-CNNs (Ma et al., 2018)	0.755
ORION (Sedaghat et al., 2017)	0.778
LightNet (Zhi et al., 2017)	**0.798**
JointNet (Ours)	0.749

Table 2
Performance of different methods on three datasets.

Method	HDRObject9 (%)			SMDObject6 (%)			SUObjects (%)		
	Precision	Recall	F1 score	Precision	Recall	F1 score	Precision	Recall	F1 score
HD (Bertrand, 2009)	80.0	80.8	80.4	66.5	68.2	65.6	54.7	59.8	53.6
SI (Johnson and Hebert, 1999)	84.5	84.3	84.4	71.1	71.4	69.9	55.5	65.3	58.2
SHOT (Tombari et al., 2010)	88.3	86.4	87.3	77.2	77.4	77.1	55.2	54.6	51.6
PointNet (Qi et al., 2016)	92.6	92.4	92.5	89.5	83.2	83.4	67.1	58.1	58.7
JointNet (Ours)	**95.9**	**95.7**	**95.8**	**94.7**	**91.0**	**92.3**	**70.3**	**76.9**	**70.7**

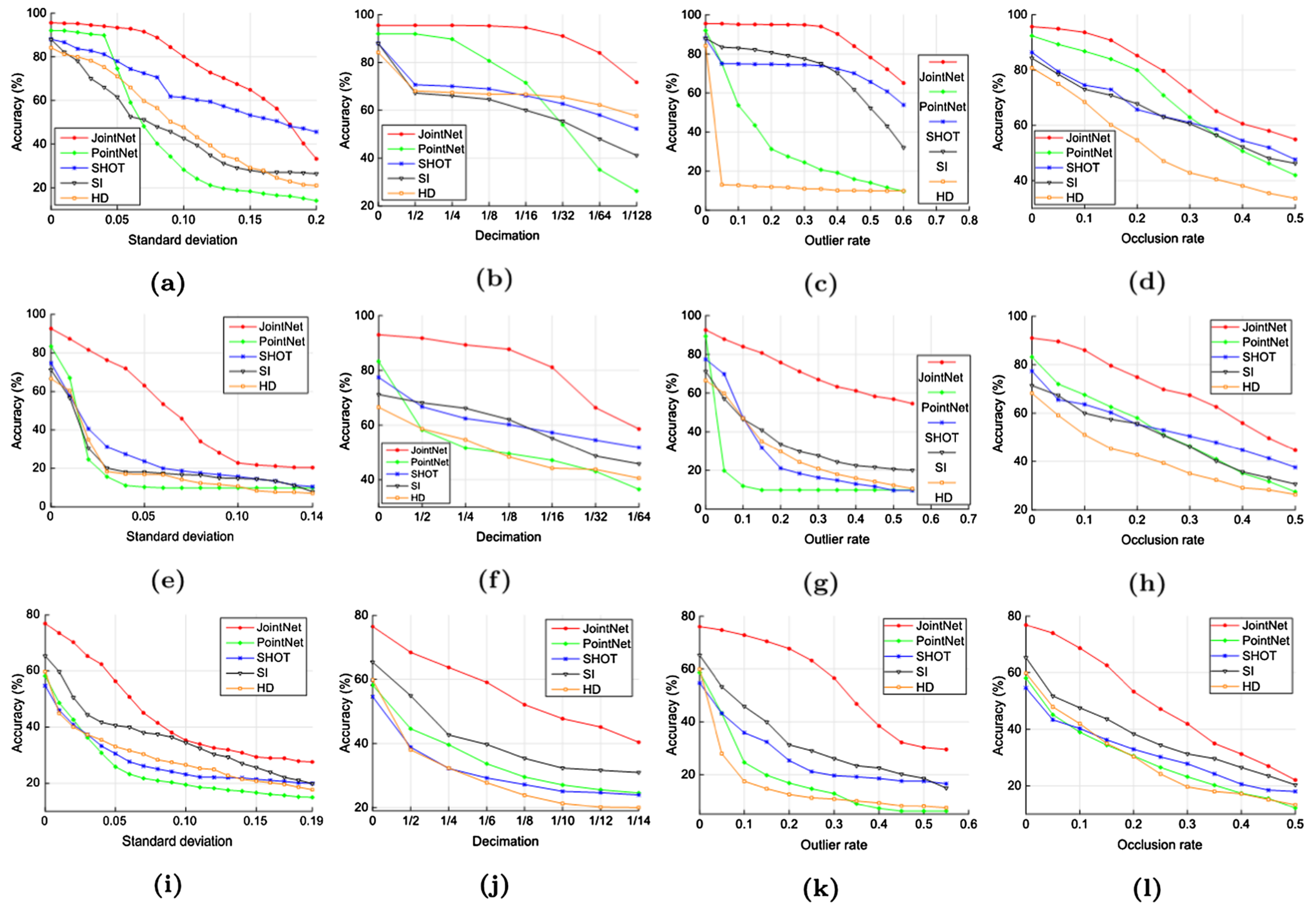


Fig. 12. Comparison of the change curves of accuracy generated by selected methods with respect to Gaussian noise, down-sampling, outliers and occlusion on three datasets. (a)–(d) HDRObject9, (e)–(h) SMDObject6, (i)–(l) Sydney Urban Objects, respectively.

$$\text{occlusion rate} = \frac{\text{volume of occluded space}}{\text{volume of normalized space}} = \frac{\text{volume of occluded space}}{8} \tag{18}$$

Considering the general situation of objects being occluded under the road scene, we take the $z = 2$, $y = 2$ for the occluded space, and so the occlusion rate is:

$$\text{occlusion rate} = \frac{x \cdot 2 \cdot 2}{8} = \frac{x}{4} \tag{19}$$

where x, y, z are the length, width and height of occluded space, respectively. Then, we took different levels of occlusion disturbance on three datasets. Fig. 12(d), (h) and (l) show the compared results. As shown in Fig. 12(d), our method achieves excellent accuracy, which is above 80%, when the occlusion rate is less than 0.2. As the occlusion rate further increases, the performance deteriorates. On the other two datasets, as shown in Fig. 12(h) and (l), our method is somewhat sensitive to occlusion, but it still outperforms other methods by a significant margin.

In summary, all these results indicate that our method is more robust to disturbance than other selected methods. There are two main reasons. First, different from handcrafted-based methods that insufficiently describing 3D data using statistical information, JointNet utilizes the deep neural network to learn high-level features, which offer more comprehensive and robust information for recognition task. Second, compared to PointNet learning deep features without leveraging local geometric structure, our method fuses the local and global distribution maps of points generated by three designed feature descriptors. This further improves the robustness and distinctiveness.

Training with disturbances. The possible disturbances in data preprocessing would deteriorate the performance of JointNet. Therefore, we tested the robustness of JointNet to the training dataset with disturbances. Considering the practical application, we take two disturbances into account. The first one is the normal noises, including Gaussian noise, down sampling, outliers and occlusion. Second one is the noisy label data, which is not only a common disturbance in preprocessing, but also a hot topic in computer vision (Xiao et al., 2015; Jiang et al., 2018).

We firstly evaluated the normal noises. Accuracy was used to measure the performance. The results are presented in Fig. 13. It is clear that our method achieves better performance with down sampling than other disturbances. Specifically, as shown in Fig. 13(b), the test accuracy is still higher than 80%, even though the decimation rate is 1/16. This demonstrates that JointNet is robust to training data with down sampling. Additionally, as shown in Fig. 13(c) and (d), JointNet performs somewhat robustly to outlier and occlusion. For the outlier disturbance, when the rate is less than 0.2, the accuracy is higher than 85% on HDRObject9 and 70% on SMDObject6. For the occlusion, the accuracy is higher than 90% on HDRObject9 when the occlusion rate is less than 0.1.

These results are attributable to the following factors. First, three designed feature descriptors preserve the distribution of points robustly, so even the training samples are disturbed by noise, the feature images generated by these three descriptors can still capture accurate geometric structure information. Second, the two-stage fusion network increases the stability and robustness of JointNet to training samples with noise, especially the occlusion and outliers.

However, JointNet may not perform well on training data with

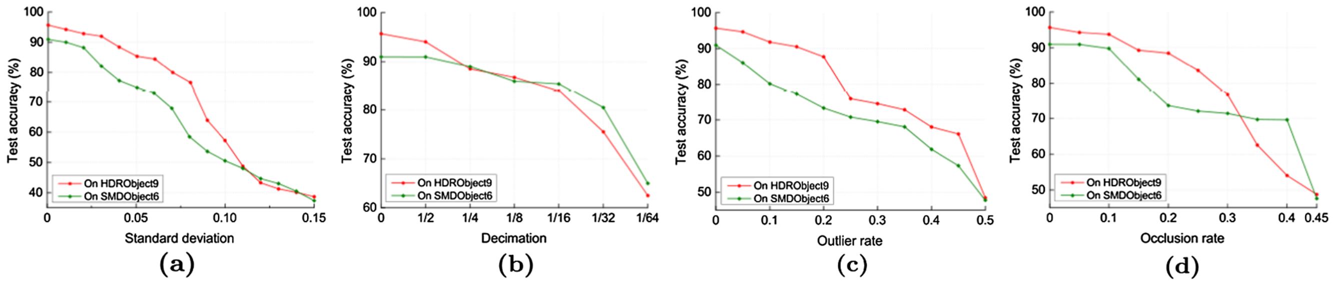


Fig. 13. Change curves of test accuracy generated by JointNet on training samples with different disturbances. (a)–(d) Gaussian noise, down-sampling, outliers and occlusion, respectively. (red: results on HDRObject9, green: results on SMDObject6.) (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Gaussian noise (Fig. 13(a)). The test accuracy is higher than 70% when the standard deviation is less than 0.05. However, the performance deteriorates as the standard deviation further increases. The reason may lie in that the geometric structure would be seriously damaged after adding Gaussian noise, and so the 3D shape is seriously deformed.

We also evaluated the effect of combining different kinds of noises on recognition task. PR curve was used to measure the performance. We denote $C = (\delta, \eta, \lambda, \gamma)$ as the combination of the noise parameters, where δ, η, λ and γ represent the standard deviation of Gaussian noise, down-sampling ratio, outlier rate, and occlusion rate, respectively. Fig. 14(a) and (b) show the results on HDRObject9 and SMDObject6, respectively. It can be found that JointNet achieves excellent performance when the disturbance is not serious, especially with the combination C = (0.01, 1/2, 0.05, 0.05) (red curve), on both two datasets. However, the recall and precision of JointNet deteriorate when these combined noises become more severe. The reason may be that the combined noise would deform the 3D shape seriously and so the training data could not provide rich and accurate geometric structure information.

Finally, we evaluated the noisy labeled training data. In this experiment, since the batch size was 32, the noisy labeled data rate ranged from 1/32 to 8/32. Accuracy was used as the metric. Fig. 14(c) shows the accuracy curves tested on HDRObject9 and SMDObject6. It is observed that the test accuracy of JointNet is higher than 85% when the rate is less than 2/32 on both two datasets, but it decreases significantly as the rate continues increasing. This indicates that the noisy labeled training data will deteriorate the performance of JointNet, and so more attention should be took in data preprocessing.

4.6. Efficiency

To demonstrate the time and memory efficiency of our method, we conducted a thorough evaluation. Specially, for handcrafted-feature-based methods, we calculated the dimension and memory footprint of descriptors, and the time for recognition on both HDRObject9 and SMDObject6. For DL-based methods, we ran 1000 forward-backward iterations, and then calculated the average time. In addition, we evaluated the model size and recorded the peak GPU memory consumption, and also calculated the time for recognition on both HDRObject9 and SMDObject6.

Table 4 shows the comparison results. Our method is about 20 ms and three times faster than PointNet on backward and forward operations, respectively. This is because in JointNet, the number of convolution layers is less than that in PointNet, and so JointNet takes much less time to compute gradients. However, PointNet has a smaller model size and GPU memory consumption, while our method consumes more memory. The reason is that most of the sizes of the convolution kernels used in PointNet are 1 × 1, while our method uses the 5 × 5 and 3 × 3 convolution kernels.

Additionally, the last two columns of Table 4 show comparison results on the recognition time. It is clear that PointNet achieves the best performance on both HDRObject9 and SMDObjcet6, while handcrafted-feature-based methods run much slower. Our method is more efficient over handcrafted-feature-based methods. For example, on HDRObject9, JointNet is more than five times, eight times and fifty-nine times faster than the HD, SI and SHOT, respectively. And on SMDObject6, JointNet is about seven times, ten times and fifty times faster than the HD, SI and SHOT, respectively. However, our method is still slower than PointNet. Especially on the HDRObject9, PointNet is about 7 s faster than JointNet. The reason is that our method needs time to compute the three feature descriptors (HSSI, VQAI and VAAI), while PointNet processes the raw point clouds directly. In this experiments, the time of calculating three descriptors on HDRObject9 and SMDObjcet6 is about 20.43 s and 4.8 s, respectively. This indicates that the time of inference on HDRObject9 and SMDObject6 is just about 8.78 s and 2.89 s, respectively.

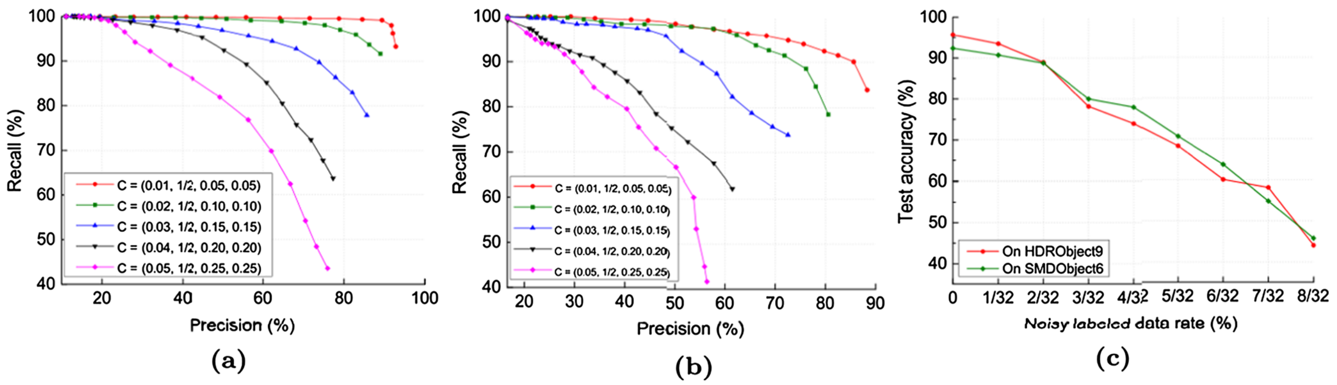


Fig. 14. Performance of JointNet on training samples with different disturbances. (a) PR-curves on HDRObject9. (b) PR-curves on SMDObject6. (c) Test accuracy of JointNet on two datasets with different noisy labeled data rates. (red: results on HDRObject9, green: results on SMDObject6.) (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 4
Space and time complexity of different methods for recognition. Model size and GPU-memory consumption are used to evaluate the space complexity. The time of forward-backward and recognition are used to evaluate the efficiency. G-M denotes the GPU-memory consumption. BW, FW, H9 and S6 stand for the backward, forward, HDRObjcet9 and SMDObject6, respectively.

Method	Dimension (float)	Memory footprint (byte)	Model size (MB)	GPU-M (MB)	Time of BW (ms)	Time of FW (ms)	Time on H9 (s)	Time on S6 (s)
HD (Bertrand, 2009)	100	400	–	–	–	–	173	57
SI (Johnson and Hebert, 1999)	153	612	–	–	–	–	259	79
SHOT (Tombari et al., 2010)	352	1408	–	–	–	–	1730	391
PointNet (Charles et al., 2016)	–		42	2842	120.62	88.05	22.81	7.26
JointNet	–		223	15,647	109.13	33.91	29.21	7.69

In summary, the handcrafted-feature-based methods take much time in calculating the descriptors, which rely heavily on the local surface neighborhoods, therefore, they have low efficiency. For DL-based methods, since PointNet processes the raw point clouds directly, it has the advantage on the time. However, because of the large number of convolution layers in the PointNet, it performs much slower on backward and forward operations than our method.

5. Discussion

As shown in Section 4, the proposed method achieves a better performance than other selected approaches on descriptiveness, robustness and efficiency. Such superior performance may come from the following factors: (1) The three feature descriptors, i.e. HSSI, VQAI, and VAAI, can robustly preserve the spatial relationship of points from three different perspectives, which would help to capture the basic geometric information. (2) By fusing the designed feature descriptors, our method can learn the high-level features, which is demonstrated to help enhance the descriptiveness.

For the handcrafted-feature-based methods, such as SI, SHOT and HD, the main issue lies in insufficiently describing 3D data using statistical information. This makes the descriptiveness still far from satisfactory. Compared with these methods, our approach makes full use of the deep neural network to learn the high-level features. More specifically, the two-stage fusion framework combines well-engineered CNNs and three proposed feature descriptors to mine more descriptive information. Therefore, JointNet can be applied in describing complex 3D shape.

For PointNet, it achieves impressive performance on synthetic datasets, such as ModelNet40. However, because PointNet processes each point in a 3D shape individually, the geometric relationship among the neighboring points would not be well employed. Therefore, PointNet would be weak in descriptiveness and sensitive to disturbances such as Gaussian noise, down sampling, outliers and occlusion, which are all unavoidable in a real scene, especially in road environments. In contrary, JointNet takes feature maps generated by the proposed three descriptors as input. Because these descriptors preserve the spatial relationship of points from different perspectives, local geometric structure information among neighboring points would be encoded in the feature maps. Consequently, these maps would improve the performance of JointNet.

Limitations. Firstly, as shown in Table 4, the model size of JointNet is large. This will weaken its generalization ability. Secondly, as discussed in Section 4.5, although JointNet is robust to data with noises, such as Gaussian noise, down sampling and outliers, it is still somewhat sensitive to noisy labeled data. Finally, compared with PointNet, JointNet requires extra time to calculate the feature descriptors. These shortcomings show clearly the direction of our future work.

6. Conclusions

We have presented a new framework, JointNet, by jointing low-level features and CNNs for 3D object recognition, and evaluated the descriptiveness, robustness and efficiency of JointNet on three real road-scene datasets, including the public dataset, the Sydney Urban Objects, and two datasets collected by a MLS system. In conclusion, our proposed method has several major advantages: First, the proposed three designed feature descriptors can learn spatial relationships of points from three different perspectives, providing rich basic geometric information. Second, JointNet uses the CNNs as the feature detector, so the high-level semantic information can be extracted to improve the descriptiveness. Third, our proposed method is robust to noise, outliers, occlusion and varying point density. Thus, JointNet is suitable for practical applications, such as automatic driving. Last, comparing with DL-based methods, such as PointNet, JointNet is more time-saving in forward-backward operations and so can speed up the training speed. Comparative experiments clearly demonstrate that JointNet outperforms the other selected methods by a large margin in terms of recognition error and computational time. It can be concluded that our method can recognize 3D MLS point clouds under road scene more accurately, robustly and efficiently.

In the future, we plan to refine our work in the following aspects: reducing the number of model parameters in an effective way to further improve the generalization ability; considering more low-level features, such as the intensity and RGB values, to improve descriptiveness; designing more suitable and effective fusion strategy to exploit the representation.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under grants of 41871380, 41471379, 61371144 and U1605254, and the Natural Sciences and Engineering Research Council of Canada under a grant of 50503-10284.

References

Bai, S., Bai, X., Zhou, Z., Zhang, Z., Latecki, L.J., 2016. GIFT: a real-time and scalable 3D shape search engine. In: IEEE Conference on Computer Vision and Pattern Recognition. IEEE, Las Vegas, Nevada, pp. 5023–5032.
Bertrand, D., 2009. Laser and vision based classification in urban environments. Ph.D. Thesis. University of Sydney, Australia.
Boulch, A., Guerry, J., Saux, B.L., Audebert, N., 2018. SnapNet: 3D point cloud semantic labeling with 2D deep segmentation networks. Comput. Graphics 71, 189–198.
Broggi, A., Buzzoni, M., Debattisti, S., Grisleri, P., Laghi, M.C., Medici, P., Versari, P., 2013. Extensive tests of autonomous driving technologies. IEEE Trans. Intell. Transport. Syst. 14 (3), 1403–1415.
Chen, T., Dai, B., Liu, D., Song, J., 2014. Performance of global descriptors for velodyne-based urban object recognition. In: IEEE Intelligent Vehicles Symposium. IEEE, Dearborn, Michigan, pp. 667–673.
Chen, X., Ma, H., Wan, J., Li, B., Xia, T., 2016. Multi-view 3D object detection network for autonomous driving. In: IEEE Conference on Computer Vision and Pattern Recognition. IEEE, Las Vegas, Nevada, pp. 6526–6534.
Davis, J., Goadrich, M., 2006. The relationship between precision-recall and ROC curves. In: International Conference on Machine Learning, Pittsburgh Pennsylvania, pp. 233–240.
Deuge, M.D., Quadros, A., Hung, C., Douillard, B., 2013. Unsupervised feature learning for classification of outdoor 3D scans. In: Australasian Conference on Robotics and Automation. ARAA, Sydney, Australia, pp. 1097–1105.
Dong, Z., Yang, B., Hu, P., Scherer, S., 2018. An efficient global energy optimization approach for robust 3D plane segmentation of point clouds. ISPRS J. Photogrammetry Remote Sens. 137, 112–133.

Dong, Z., Yang, B., Liu, Y., Liang, F., Li, B., Zang, Y., 2017. A novel binary shape context for 3D local surface description. ISPRS J. Photogrammetry Remote Sens. 130, 431–452.

Engelmann, F., Kontogianni, T., Hermans, A., Leibe, B., 2017. Exploring spatial context for 3D semantic segmentation of point clouds. In: IEEE International Conference on Computer Vision. IEEE, Venice, Italy, pp. 716–724.

Frome, A., Huber, D., Kolluri, R., Bulow, T., Malik, J., 2004. Recognizing objects in range data using regional point descriptors. In: European Conference on Computer Vision. Springer, Prague, Czech Republic, pp. 224–237.

Girshick, R., 2015. Fast R-CNN. In: IEEE International Conference on Computer Vision. IEEE, Santiago, Chile, pp. 1440–1448.

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. Photogrammetry Remote Sens. 87 (1), 93–107.

Guo, Y., Bennamoun, M., Sohel, F., Lu, M., Wan, J., 2014. 3D object recognition in cluttered scenes with local surface features: a survey. IEEE Trans. Pattern Anal. Mach. Intell. 36 (11), 2270–2287.

Guo, Y., Bennamoun, M., Sohel, F., Lu, M., Wan, J., Kwok, N.M., 2016. A comprehensive performance evaluation of 3D local feature descriptors. Int. J. Comput. Vision 116 (1), 66–89.

Guo, Y., Sohel, F., Bennamoun, M., Lu, M., Wan, J., 2013a. Rotational projection statistics for 3D local surface description and object recognition. Int. J. Comput. Vision 105 (1), 63–86.

Guo, Y., Sohel, F., Bennamoun, M., Lu, M., Wan, J., 2013b. TriSI: a distinctive local surface descriptor for 3D modeling and object recognition. In: International Conference on Computer Graphics Theory and Applications, pp. 86–93.

He, K., Gkioxari, G., Dollr, P., Girshick, R., 2017. Mask R-CNN. In: IEEE International Conference on Computer Vision. IEEE, Venice, Italy, pp. 2980–2988.

He, K., Zhang, X., Ren, S., Sun, J., 2015. Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans. Pattern Anal. Mach. Intell. 37 (9), 1904–1916.

Hu, P., Dong, Z., Yuan, P., Liang, F., Yang, B., 2018. Reconstruction of 3D models from point clouds with hybrid representation. ISPRS Archives XLII-2, 449–454.

Ioannidou, A., Chatzilari, E., Nikolopoulos, S., Kompatsiaris, I., 2017. Deep learning advances in computer vision with 3D data: a survey. ACM, Computing Surveys. https://doi.org/10.1145/3042064.

Jiang, L., Zhou, Z., Leung, T., Li, L.J., Li, F.-F., 2018. MentorNet: learning data-driven curriculum for very deep neural networks on corrupted labels. International Conference on Machine Learning. Stockholm, Sweden. arXiv:1712.05055.

Johnson, A.E., Hebert, M., 1999. Using spin images for efficient object recognition in cluttered 3D scenes. IEEE Trans. Pattern Anal. Mach. Intell. 21 (5), 433–449.

Klokov, R., Lempitsky, V., 2017. Escape from cells: Deep kd-networks for the recognition of 3D point cloud models. In: IEEE International Conference on Computer Vision. IEEE, Venice, Italy, pp. 863–872.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In: Annual Conference on Neural Information Processing Systems, Lake Tahoe UT, pp. 1097–1105.

Laptev, D., Savinov, N., Buhmann, J.M., Pollefeys, M., 2016. TI-POOLING: transformation invariant pooling for feature learning in convolutional neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition. IEEE, Las Vegas, Nevada, pp. 289–297.

Lecun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521 (7553), 436.

Ma, C., Guo, Y., Lei, Y., An, W., 2018. Binary volumetric convolutional neural networks for 3-D object recognition. IEEE Trans. Instrum. Meas. 68 (1), 38–48.

Maturana, D., Scherer, S., 2015. VoxNet: a 3D convolutional neural network for real-time object recognition. In: IEEE International Conference on Intelligent Robots and Systems. IEEE, Hamburg, Germany, pp. 922–928.

Qi, C.R., Su, H., Mo, K., Guibas, L.J., 2016. PointNet: deep learning on point sets for 3D classification and segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition. IEEE, Las Vegas, Nevada, pp. 77–85.

Qi, C.R., Yi, L., Su, H., Guibas, L.J., 2017. PointNet++: deep hierarchical feature learning on point sets in a metric space. In: Annual Conference on Neural Information Processing Systems, Los Angeles, California, pp. 77–85.

Qin, N., Hu, X., Dai, H., 2018. Deep fusion of multi-view and multimodal representation of ALS point cloud for 3D terrain scene recognition. ISPRS J. Photogrammetry Remote Sens. 143, 205–212.

Ren, S., Girshick, R., Girshick, R., Sun, J., 2015. Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. 39 (6), 1137–1149.

Riegler, G., Ulusoy, A.O., Geiger, A., 2017. OctNet: learning deep 3D representations at high resolutions. In: IEEE Conference on Computer Vision and Pattern Recognition. IEEE, Honolulu, Hawaii, pp. 3577–3586.

Rusu, R.B., Blodow, N., Beetz, M., 2009. Fast point feature histograms (FPFH) for 3D registration. In: IEEE International Conference on Robotics and Automation. IEEE, Kobe, Japan, pp. 3212–3217.

Salti, S., Tombari, F., Stefano, L.D., 2014. SHOT: unique signatures of histograms for surface and texture description. Comput. Vision Image Understanding 125 (8), 251–264.

Schreiber, M., Knppel, C., Franke, U., 2013. LaneLoc: lane marking based localization using highly accurate maps. In: IEEE Intelligent Vehicles Symposium. IEEE, Gold Coast City, Australia, pp. 449–454.

Sedaghat, N., Zolfaghari, M., Amiri, E., Brox, T., 2017. Orientation-boosted voxel nets for 3D object recognition. British Machine Vision Conference, London, UK. arXiv:1604.03351.

Seo, Y.W., Lee, J., Zhang, W., Wettergreen, D., 2015. Recognition of highway workzones for reliable autonomous driving. IEEE Trans. Intell. Transport. Syst. 16 (2), 708–718.

Shah, S.A.A., Bennamoun, M., Boussaid, F., 2017. Keypoints-based surface representation for 3D modeling and 3D object recognition. Pattern Recognit. 64, 29–38.

Su, H., Maji, S., Kalogerakis, E., Learnedmiller, E., 2015. Multi-view convolutional neural networks for 3D shape recognition. In: IEEE International Conference on Computer Vision. IEEE, Santiago, Chile, pp. 945–953.

Sukno, F.M., Waddington, J.L., Whelan, P.F., 2013. Rotationally invariant 3D shape contexts using asymmetry patterns. In: International Conference on Computer Graphics Theory and Applications. ACM, Barcelona, Spain, pp. 7–17.

Tatarchenko, M., Dosovitskiy, A., Brox, T., 2017. Octree generating networks: efficient convolutional architectures for high-resolution 3D outputs. In: IEEE International Conference on Computer Vision. IEEE, Venice, Italy, pp. 2107–2115.

Tombari, F., Salti, S., DiStefano, L., 2013. Performance evaluation of 3D keypoint detectors. Int. J. Comput. Vision 102 (1), 198–220.

Tombari, F., Salti, S., Stefano, L.D., 2010. Unique shape context for 3D data description. In: ACM Workshop on 3D Object Retrieval. ACM, Norrkoping, Sweden, pp. 57–62.

Wang, P.S., Liu, Y., Guo, Y.X., Sun, C.Y., Tong, X., 2017. OCNN: octree-based convolutional neural networks for 3D shape analysis. ACM Trans. Graphics. https://doi.org/10.1145/3072959.3073608.

Wen, C., Pan, S., Wang, C., Li, J., 2016. An indoor backpack system for 2-D and 3-D mapping of building interiors. IEEE Geosci. Remote Sens. Lett. 13 (7), 992–996.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J., 2014. 3D ShapeNets: a deep representation for volumetric shapes. In: IEEE Conference on Computer Vision and Pattern Recognition. IEEE, Columbus, Ohio, pp. 1912–1920.

Xiao, T., Xia, T., Yang, Y., Huang, C., 2015. Learning from massive noisy labeled data for image classification. In: IEEE Conference on Computer Vision and Pattern Recognition. IEEE, Las Vegas, Nevada, pp. 2691–2699.

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

Yang, B., Liu, Y., Dong, Z., Liang, F., Li, B., Peng, X., 2017. 3D local feature BKD to extract road information from mobile laser scanning point clouds. ISPRS J. Photogrammetry Remote Sens. 130, 329–343.

Yu, Y., Li, J., Guan, H., Jia, F., Wang, C., 2015a. Three-dimensional object matching in mobile laser scanning point clouds. IEEE Geosci. Remote Sens. Lett. 12 (3), 492–496.

Yu, Y., Li, J., Guan, H., Jia, F., Wang, C., 2017. Learning hierarchical features for automated extraction of road markings from 3-D mobile LiDAR point clouds. IEEE J. Sel. Top. Appl. Earth Observations Remote Sens. 8 (2), 709–726.

Yu, Y., Li, J., Guan, H., Wang, C., 2015b. Automated extraction of urban road facilities using mobile laser scanning data. IEEE Trans. Intell. Transport. Syst. 16 (4), 2167–2181.

Yu, Y., Li, J., Guan, H., Wang, C., 2016. Automated detection of three-dimensional cars in mobile laser scanning point clouds using dbm-hough-forests. IEEE Trans. Geosci. Remote Sens. 54 (7), 4130–4142.

Yu, Y., Li, J., Guan, H., Wang, C., Yu, J., 2015c. Semiautomated extraction of street light poles from mobile LiDAR point-clouds. IEEE Trans. Geosci. Remote Sens. 53 (3), 1374–1386.

Zai, D., Li, J., Guo, Y., Cheng, M., Huang, P., Cao, X., Wang, C., 2017. Pairwise registration of TLS point clouds using covariance descriptors and a non-cooperative game. ISPRS J. Photogrammetry Remote Sens. 134, 15–29.

Zhi, S., Liu, Y., Li, X., Guo, Y., 2017. LightNet: a lightweight 3D convolutional neural network for real-time 3D object recognition. In: Eurographics Workshop on 3D Object Retrieval London, UK, pp. 9–16.