Human-centric Computing and Information Sciences

RESEARCH Open Access

CNN-based 3D object classification using Hough space of LiDAR point clouds

Wei Song[1,3*], Lingfeng Zhang[1], Yifei Tian[2], Simon Fong[2], Jinming Liu[1] and Amanda Gozho[1]

*Correspondence: sw@ncut.edu.cn
[1] School of Information Science and Technology, North China University of Technology, Beijing, China
Full list of author information is available at the end of the article

Abstract
With the wide application of Light Detection and Ranging (LiDAR) in the collection of high-precision environmental point cloud information, three-dimensional (3D) object classification from point clouds has become an important research topic. However, the characteristics of LiDAR point clouds, such as unstructured distribution, disordered arrangement, and large amounts of data, typically result in high computational complexity and make it very difficult to classify 3D objects. Thus, this paper proposes a Convolutional Neural Network (CNN)-based 3D object classification method using the Hough space of LiDAR point clouds to overcome these problems. First, object point clouds are transformed into Hough space using a Hough transform algorithm, and then the Hough space is rasterized into a series of uniformly sized grids. The accumulator count in each grid is then computed and input to a CNN model to classify 3D objects. In addition, a semi-automatic 3D object labeling tool is developed to build a LiDAR point clouds object labeling library for four types of objects (wall, bush, pedestrian, and tree). After initializing the CNN model, we apply a dataset from the above object labeling library to train the neural network model offline through a large number of iterations. Experimental results demonstrate that the proposed method achieves object classification accuracy of up to 93.3% on average.

Keywords: 3D object classification, LiDAR point clouds, Hough space, CNN

Introduction

Object classification is a significant research task in artificial intelligence (AI) and computer vision domain and has been applied to scene understanding, domestic/service robots, and smart factory systems [1, 2]. For autonomous driving, high-precision object classification results enable an intelligent driving system to identify obstacles and achieve safe autonomous route planning [3]. High-precision object classification results are vital as a preliminary step for subsequent work.

Light detection and ranging (LiDAR) sensors have been employed increasingly for object classification because such sensors can obtain a large amount of high-resolution and accurate 3D point clouds from the surrounding environment [4, 5]. Traditional classification methods from point clouds primarily analyze and extract features, such as geometric attributes, shape attributes, or structural attributes, and then classify objects by training a model [6, 7]. However, the disordered arrangement, inhomogeneous densities, and non-structural distribution characteristics of LiDAR point clouds tend to affect classification accuracy and speed performance [8]. Therefore, 3D object classification based on LiDAR point clouds remains a challenging problem.

Deep learning methods have been widely studied and demonstrate the most advanced 3D object classification performance, especially Convolutional Neural Networks (CNNs) [9]. For weight sharing and kernel function optimization, traditional CNNs require regular data structures as input; thus, the point cloud is typically processed with multi-view or voxel and then input into deep network [10]. However, this process typically causes problems, such as geometric structure loss and resolution reduction [11]. In addition, a trend in 3D object classification is to mix different representations of point clouds and CNN models to generate a sufficient amount of discriminating information about the objects [12].

This paper proposes a CNN-based 3D object classification method using the Hough space of LiDAR point clouds. The initialized CNN model is trained based on all grids' accumulator counts, which are generated using a projection of the 3D points into Hough space and rasterization. In addition, due to a lack of open training datasets, a semi-automatic 3D object labeling tool is developed to divide LiDAR point clouds into four object types, i.e., wall, bush, pedestrian, and tree, to train and test the proposed CNN model.

The remainder of this paper is organized as follows. Section "Related works" provides an overview of related work. Section "CNN-based 3D object classification from Hough Space" describes the proposed method. Section "Experiments and analysis" describes the experimental procedures and evaluates classification results. Finally, Section "Conclusions" concludes the paper.

Related works

With its outstanding advantages over traditional digital cameras, LiDAR can acquire highly accurate 3D point clouds regardless of illumination, shadow, and texture [13]. LiDAR is used in a wide range of applications, such as semantic environment perception, 3D environment reconstruction, and automatic navigation. Therefore, object classification from LiDAR point clouds has received increasing attention and has a promising future.

Traditionally, object classification methods have been divided into global feature-based and local feature-based methods. In global feature-based methods, point clouds are first pre-segmented, and potential objects are divided into clusters. Then, researchers defined a set of global features and identified the objects as a whole. For example, Rusu et al. [14] proposed the view feature histogram (VFH) and added viewpoint information into the calculation of the angle between the relative normals to maintain a constant rotation scale. However, the VFH depends on only the geometric information of the entire 3D object surface and shows low accuracy when identifying objects with similar geometric information. To increase the descriptiveness of global features, Wohlkinger and Vincze [15] proposed the ensemble of shape functions descriptor, which comprises of three shape functions, i.e., distance, angle, and area distribution of the surfaces of local point clouds, into a high-performance global shape descriptor. Chen et al. [16] proposed a global Fourier histogram descriptor that uses cylindrical angular coordinate and is independent of the rotation around the vertical axis. Generally, global feature-based methods have high calculation speed and accuracy for object classification with simple shapes; however, they have insufficient descriptions of details and are sensitive to both noise and occlusion.

In local feature-based methods, first, the key points of the scene are extracted directly, and then the spatial distribution or geometric properties are computed in the neighborhood of each key point to forge local features. Zhu et al. [17] classified the local shapes of point clouds using the surface shape description method to obtain candidate feature point areas and only reconstructed a few important feature points to avoid meaningless calculations. However, spin-image method suffers from low descriptiveness and is easily affected by density. To overcome the low descriptiveness problem, Dong et al. [18] proposed a 3D binary shape context (BSC) descriptor that is highly efficient and descriptive. This method encodes point density and distance from three orthogonal projection planes to form abundant local surface information. In addition, their method calculates weighted projection features using Gaussian kernel density estimation. Salti et al. [19] mixed the signature and histogram structure to generate a signature of histograms of orientation (SHOT). However, these methods suffer from non-uniqueness and low accuracy. Guo et al. [20] proposed the Tri-Spin-Image local shape descriptor, which can effectively classify objects in the presence of clutter and occlusion. Prkahyas et al. [21] transformed the SHOT descriptor into a binary representation, which they called the binary signature of histograms of orientation (B-SHOT). Compared to SHOT, B-SHOT is six times faster but requires 32 times more memory. These methods are highly descriptive regardless of noise, occlusion, and clutter; however, they incur heavy computational burden due to the large-scale and high-capacity characteristics of LiDAR point clouds.

In machine learning, Serna et al. [22] segmented connected objects using a watershed method after filtering out ground points and noises, and then utilized the Support Vector Machine with geometric and contextual features to classify objects. Wang et al. [23] described object categories using Implicit Shape Model, and extended Hough Forest framework to classify objects. Becker et al. [24] applied two typical machine learning models, i.e., the random forest and gradient enhancement tree models, as classifiers, and combined color information when detecting semantic classes to achieve high-precision object classification. These methods can improve classifier robustness, but typically rely on manual feature extraction and off-the-shelf classifiers to predict object labels.

CNNs have better flexibility and universality than traditional machine learning methods and have realized remarkable achievements in object classification [25]. However, it is difficult to directly apply CNNs to the analysis of 3D points because 3D unstructured point clouds are irregular. Su et al. [26] utilized multiple pictures of a 3D meshed object using the multi-view method and developed CNN architecture to combine such information into a compact shape descriptor for object classification. Zhi et al. [27] proposed a lightweight volumetric CNN architecture named LightNet that realizes real-time 3D object classification by predicting both class and direction labels from full and partial shapes. Most existing volumetric 3D CNNs have very large and complex structures, which results in very high computational costs and storage requirements. Qi et al. [28] proposed a novel network structure, named PointNet, which combines disordered point clouds with deep learning methods for classification and segmentation. Li et al. [29] proposed PointCNN, which utilized X-Conv to perform X-transformation of point cloud, and then convolved on the transformed features. This method moderately solved the problem of mapping disordered and irregular point data into ordered and regular forms. Xu et al. [30] proposed SpiderCNN, a parameterized convolution filter, which makes convolution operation applicable to irregular point cloud data. These methods are inefficient in the utilization of the structural relationship between local neighborhood point pairs. Besides, due to the sparse characteristics of point clouds, large amounts of original data is lost after convolution, hence robust CNNs for 3D object classification are required.

CNN-based 3D object classification from Hough space

To overcome the above problems, the Hough space representation of LiDAR point clouds is combined with a CNN model to classify 3D objects. As shown in Fig. 1, the proposed method involves a semi-automatic object point clouds labeling system, object Hough space generation, and CNN-based 3D object classification.

In the proposed method, noisy and ground points, which typically affect classification accuracy and result in high computation costs, are filtered out first to eliminate interference. All non-ground points are then segmented into individual clusters using an object segmentation algorithm. Additionally, a semi-automatic object point clouds labeling tool is developed to store the information of these clusters and it manually divided individual clusters into four types of objects: wall, bush, pedestrian, and tree objects.

LiDAR point clouds have disordered arrangement and non-structural distributions; thus, the point storage order in memory is uncertain, which affects classification accuracy. To address this issue, object point clouds are projected onto x–z plane. These 2D points are transformed into Hough space using the Hough transform algorithm, which relies on the coordinate transformation principle between Cartesian coordinate and polar coordinate systems as follows:

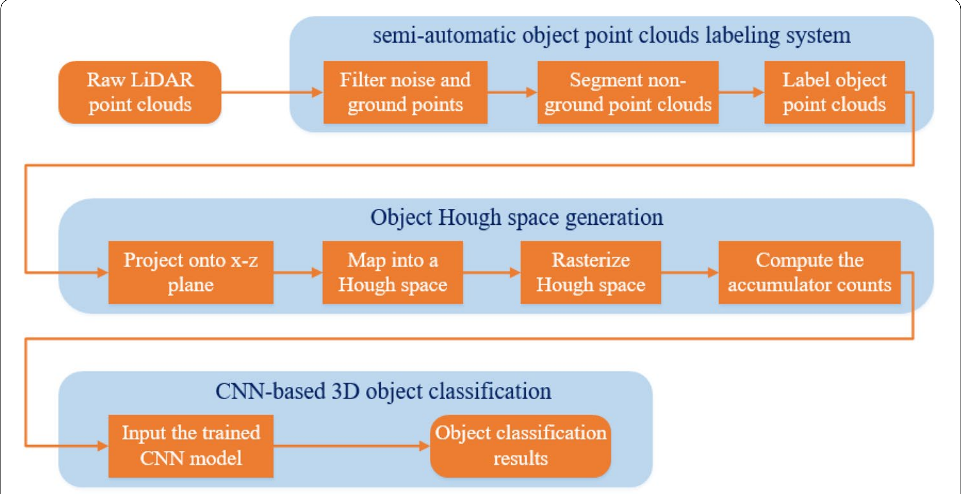
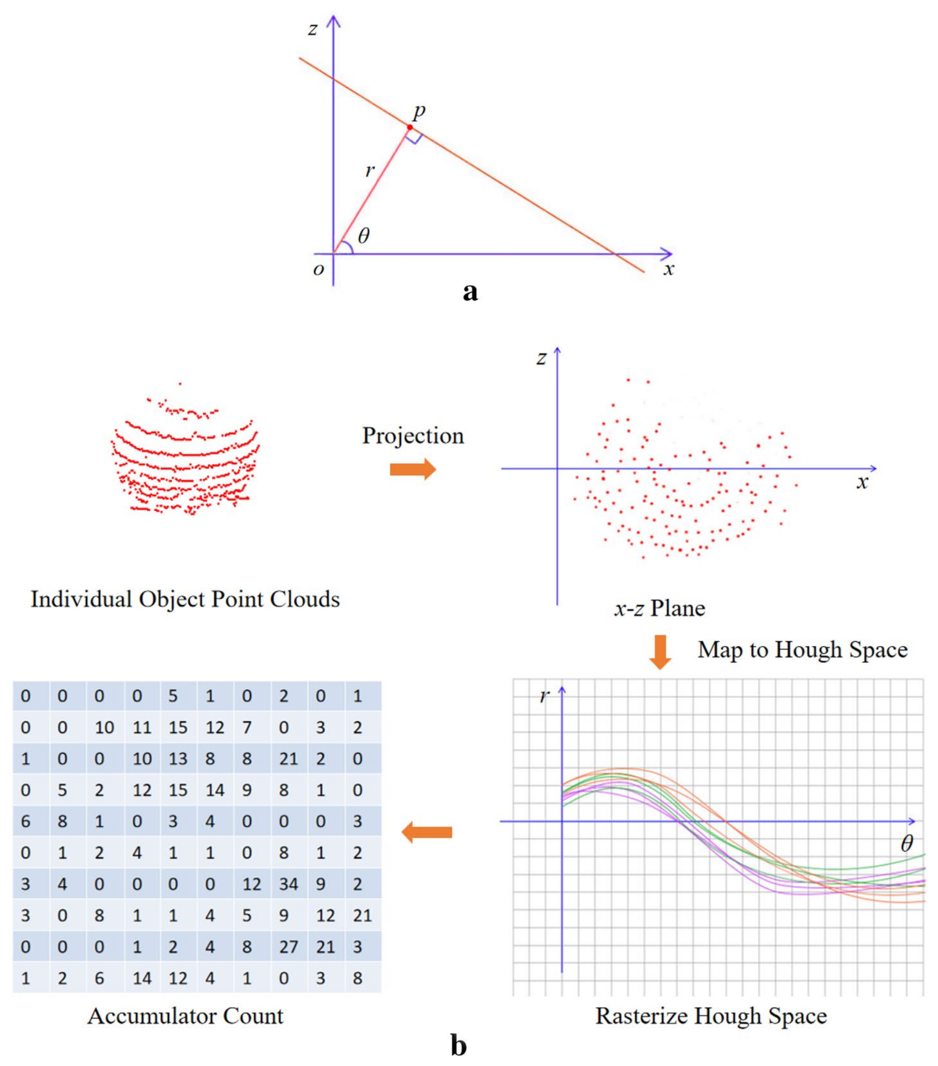

semi-automatic object point clouds labeling system
Raw LiDAR point clouds
Filter noise and ground points
Segment non-ground point clouds
Label object point clouds
Object Hough space generation
Project onto x-z plane
Map into a Hough space
Rasterize Hough space
Compute the accumulator counts
CNN-based 3D object classification
Input the trained CNN model
Object classification results

Fig. 1 Flowchart of the proposed CNN-based 3D object classification using the Hough space of LiDAR point clouds

$$r = x\cos(\theta) + z\sin(\theta) \tag{1}$$

As shown in Fig. 2 (a), variable r is the length of the line op, where o is the origin and p is a non-ground point. Variable θ is the angle between line op and the x axis. Note that the value range of r depends on the size of the collected object sample. The range of angle θ is defined as [0, π]. We generate an object Hough space $H(r,\theta)$ based on this.

As shown in Fig. 2 (b), the coordinates (x, z) of each point in Cartesian coordinates generate an individual curve in the Hough space. In the proposed method, Hough space $H(r,\theta)$ is rasterized into $m \times n$ uniform grids, and the grid resolution is defined manually according to the specific environment. Matrix A, which comprises of n rows and m columns, is applied to store the accumulator count of each grid. Subsequently, for each 2D point p_i and discrete angle θ_j, the corresponding distance r is computed using Eq. (1). The values of i and j are the indexes of 2D points and angles, respectively. The point



0	0	0	0	5	1	0	2	0	1
0	0	10	11	15	12	7	0	3	2
1	0	0	10	13	8	8	21	2	0
0	5	2	12	15	14	9	8	1	0
6	8	1	0	3	4	0	0	0	3
0	1	2	4	1	1	0	8	1	2
3	4	0	0	0	0	12	34	9	2
3	0	8	1	1	4	5	9	12	21
0	0	0	1	2	4	8	27	21	3
1	2	6	14	12	4	1	0	3	8

Fig. 2 Generated Hough space **a** Line parameters r and θ. **b** Acquiring accumulator count process

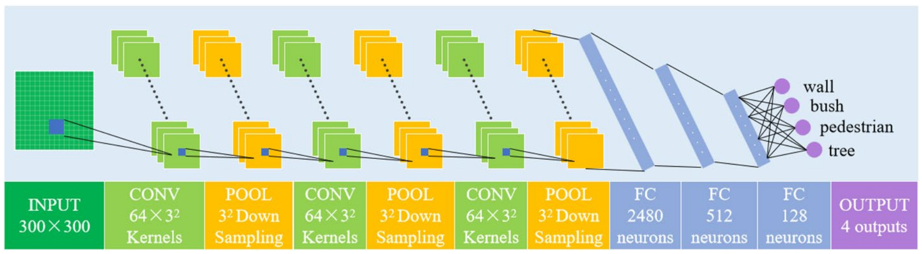


Fig. 3 Structure of the CNN model

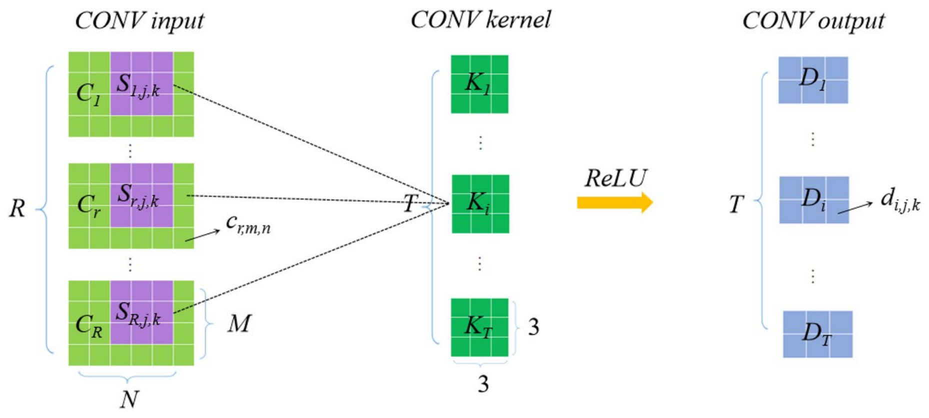


Fig. 4 Convolution processing

index i belongs to $[1, N_i]$, where N_i is the number of points in the object point clouds, and index j of angle θ is located in $[1, n]$. After traversing all angles θ_j, a series of $r_{i,j,k}$ are obtained such that the corresponding elements $a_{j,k}$ in matrix A are incremented by one. The value of k is the result of dividing r by its resolution, where $k\in[1, m]$ is defined as the index of length r. In this manner, the object Hough space generation is finished, and matrix A is updated completely when all grids have been computed.

Next, the above accumulator counts are input into a CNN model to classify objects and an eleven-layer CNN architecture is designed to adapt these data, as shown in Fig. 3. The CNN model includes a 300×300 input layer, three convolution (CONV) layers with 64 kernels of size 3×3 and a stride of 1, two pooling (POOL) layers with 3×3 down sampling, three fully-connected (FC) layers with 2480, 512, and 128 neurons, respectively, and an output layer with four outputs.

The forward-propagation mainly divides into three processes: CONV, Max-POOL, and FC. Each element $d_{i,j,k}$ of CONV output matrix D_i is computed according to the Eq. (2).

$$d_{i,j,k} = \sigma\left(\sum_{r=1}^{R}\left(S_{r,j,k} \cdot K_i\right) + b_i\right) \tag{2}$$

As shown in Fig. 4, matrices C_r and D_i are the input and output of the CONV layer, respectively. Note that each element $d_{i,j,k}$ belongs to matrix D_i. Matrix K_i is a 3×3

matrix, which is the CONV kernel. The value b_i belongs to vector B, which is the bias. The value of r is the index of the input matrix and belongs to [1, R], where R is the number of input matrices. The value of i is the index of output matrix D_i, kernel matrix K_i, and bias vector b_i. Index i is located in [1, T], where T is the number of CONV kernels. Element $c_{r,m,n}$ is a member of matrix C_r. The value of m and n are the indexes of the rows and columns of the input matrix C_r. Index m belongs to [1, M], where M is the length of input matrix C_r. Index n belongs to [1, N], where N is the width of input matrix C_r. Matrix $S_{r,j,k}$ is obtained by sampling matrix C_r with the kernels of size 3×3 and a stride of 1. The value of j and k are defined as the indexes of sampling matrix $S_{r,j,k}$, where $j \in [1, M]$ and $k \in [1, N]$. In addition, σ is the ReLU activation function.

As shown in Fig. 5, matrices P_i and E_i are the input and output of the POOL layer, respectively. The value of i is the index of the input and output matrix and belongs to [1, I], where I is the number of POOL input matrices. Matrix $Q_{i,j,k}$ is obtained by sampling matrix P_i with 3×3 down sampling. Here, $e_{i,j,k} \in E_i$ is computed using Eq. (3). The values $j \in [1, J/3]$ and $k \in [1, K/3]$ are the indexes of the rows and columns of matrix E_i, where J and K are the length and width of input matrix P_i. The function f is utilized to find the maximum value.

$$e_{i,j,k} = f(Q_{i,j,k}) \tag{3}$$

In FC processing, vector X^l and X^{l+1} are the inputs of the lth and $l+1$th layers, respectively. The values of l and $l+1$ represent the ordinal number of layers and index l belongs to [1, L], where L is the number of FC layer. The values of x_j^l and x_i^{l+1} are members of vectors X^l and X^{l+1}, respectively, and j and i are the indexes of vectors X^l and X^{l+1}. Index j belongs to [1, M], where M is the number of neurons in the lth layer. Index i belongs to [1, N], where N is the number of neurons in the $l+1$th layer. The value of $w_{ij}^l$is the weight of the ith neuron in the $l+1$th layer connected to the jth neuron in the lth layer, and the value of b_i^l is the bias of the ith neuron in the $l+1$th layer. The value of x_i^{l+1} is computed using Eq. (4).

$$x_i^{l+1} = \sigma\left(\sum_{j=1}^{M} (w_{ij}^l x_j^l) + b_i^l\right) \tag{4}$$

Next, the vector Z represents the output neurons of the output layer, and the value of z_r is a member of Z. Each element $y_r{}'$ in prediction vector Y' can be obtained by a softmax

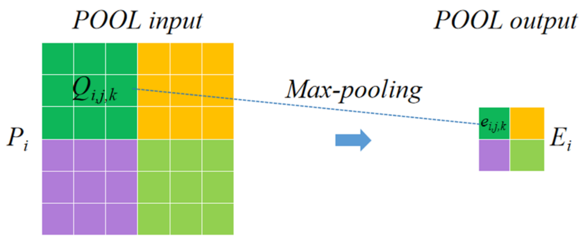


Fig. 5 Max-pooling processing

function (Eq. (5)). In addition, the value of η, which represents the error of the CNN model, is computed using Eq. (6).

$$y'_r = \frac{e^{z_r}}{\sum_{j=1}^{R} e^{z_j}} \tag{5}$$

$$\eta = -\sum_{r=1}^{R} (y_r \log(y'_r)) \tag{6}$$

Vector Y is a binary object label vector, and the value of y_r is a member of vector Y. The value of $r \in [1, R]$ is the index of vectors Z, Y', and Y, where R is the number of outputs in the output layer. The forward-propagation process is completed when the loss is obtained. Then, all CNN parameters, such as filter kernel, neuron bias, and weight, are adjusted using the Gradient descent method in the back-propagation process. Residual error δ_r^0, which is the derivative of the loss function relative to z_r, is computed using Eq. (7).

$$\delta_r^0 = \begin{cases} y'_r - 1 & y_r = 1 \\ y'_r & y_r = 0 \end{cases} \tag{7}$$

Then, the residual error δ_j^l of the jth neuron in the lth layer is computed as follows.

$$\delta_j^l = \left(\sum_{k=1}^{K} \delta_k^{l+2} w_{ki}^{l+1} \right) \sigma' \left(\sum_{j=1}^{M} (w_{ij}^l x_j^l) + b_i^l \right) \tag{8}$$

In Eq. (8), δ_k^{l+2} is defined as the kth neuron in the $l+2$th layer. Index k belongs to $[1, K]$, where K is the number of neurons in the $l+2$th layer. The value of w_{ki}^{l+1} is the weight of the kth neuron in the $l+2$th layer connected to the ith neuron in the $l+1$th layer. In addition, σ' is the derivative of the Leaky ReLU activation function. The gradient of weight w_{ij}^l and bias b_i^l are expressed follows.

$$\frac{\partial \eta}{\partial w_{ij}^l} = \delta_i^l x_j^l \tag{9}$$

$$\frac{\partial \eta}{\partial b_i^l} = \delta_i^l \tag{10}$$

As shown in Eqs. (11) and (12), weight w_{ij}^l and bias b_i^l are updated using the gradient descent method. Here, the value of α is the learning rate.

$$w_{ij}^l = w_{ij}^l - \alpha \frac{\partial \eta}{\partial w_{ij}^l} \tag{11}$$

$$b_i^l = b_i^l - \alpha \frac{\partial \eta}{\partial b_i^l} \tag{12}$$

Then, combined with the above all equations, a large number of data and iterations are applied to train the CNN model to minimize error. Finally, a testing dataset is utilized to evaluate the object classification performance of the proposed method.

Experiments and analysis

In this experiment, a LiDAR sensor (Velodyne HDL-32E) was employed to acquire high-precision 3D points from different environments. The program was executed on a 2.10 GHz Intel(R) Xeon(R) Silver 4110 CPU (with 16 GB RAM) with an NVIDIA GeForce RTX 2080 Ti GPU. The program utilized the DirectX 9.0 Software Development Kit to represent LiDAR point clouds.

Performance of semi-automatic 3D object labeling library

The generated semi-automatic 3D object labeling tool is shown in Fig. 6. The left part of the figure shows a particular 3D point cloud global scene. Here, ground points are rendered in black, and non-ground points in green. The right side shows six buttons, including two on the top for switching objects and four on the bottom for storing different object information. We classified objects into four types, i.e., wall, bush, pedestrian, and tree. When selected, an individual cluster can be rendered in red bounded by a red box.

As shown in Fig. 7, the four object types have four different spatial distributions. These individual clusters were identified manually, and the object information was stored in a 3D object dataset. This information includes raw 3D point clouds coordinates, the center point coordinates of individual objects, the coordinates relative to the center, and the object label.

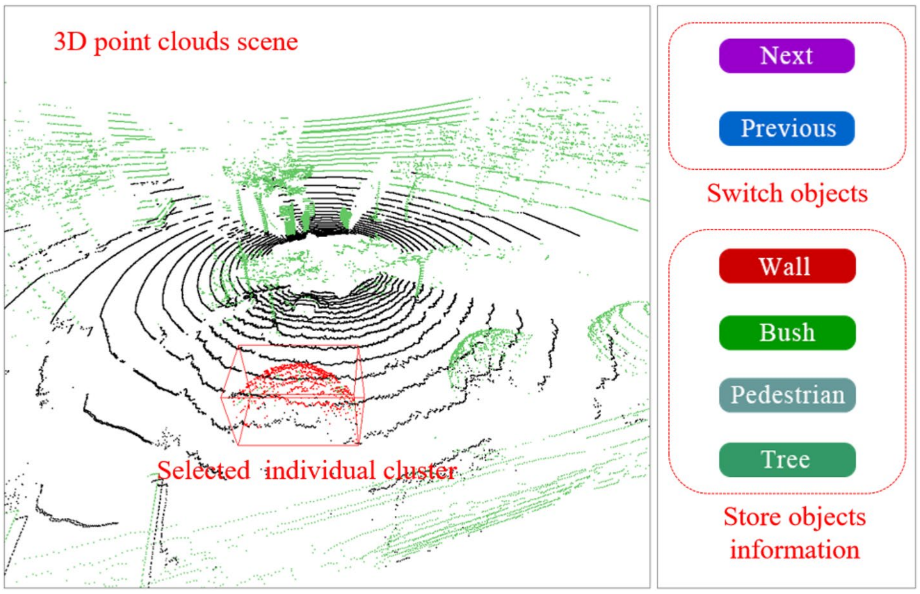


Fig. 6 Display tool of semi-automatic 3D object labeling

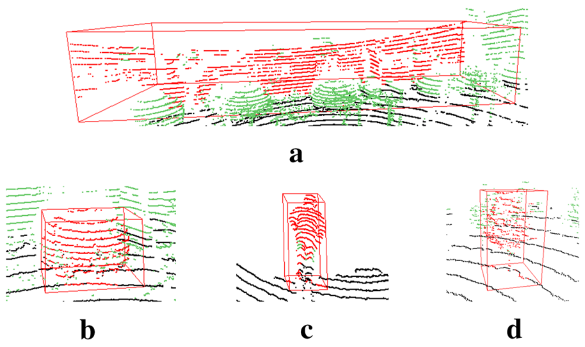


Fig. 7 Spatial distributions of 3D point clouds for four object types. **a** Wall. **b** Bush. **c** Pedestrian. **e** Tree

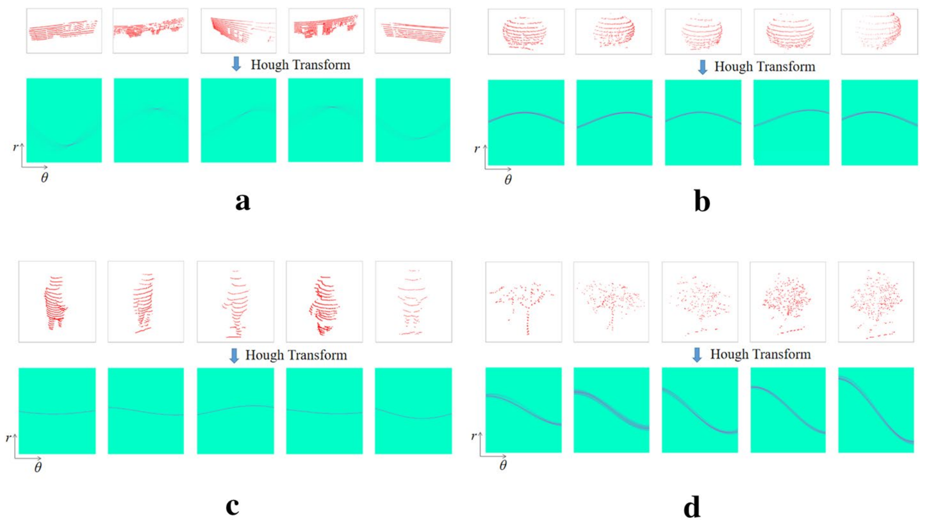


Fig. 8 Hough spaces generated by four types of object point clouds. **a** Wall Hough space. **b** Bush Hough space. **c** Pedestrian Hough space. **d** Tree Hough space

Hough space performance

In this experiment, all object point clouds were mapped into Hough space. Figure 8 shows the Hough space generated by different object point clouds respectively. Figure 8a shows images in Hough space generated by wall point clouds. Here, LiDAR point clouds were projected to the x–z plane; thus, the wall point clouds gathered into a curve that corresponds to the wall points that converged to a point in the Hough space. After being projected onto the x–z plane, bush point clouds formed some concentric circles. Figure 8b shows images in Hough space generated by the bush point clouds, which formed a series of convex curves. In contrast, when tree point clouds were projected onto the x–z plane, points near the central point were denser, and points distant from the central point were sparse. For the diversity of pedestrian morphology, there was no obvious regular distribution for the curve corresponding to pedestrians in the Hough space; however, the slope of these curves was relatively gentle and did not fluctuate significantly, as shown in Fig. 8c. Figure 8d shows images in Hough space generated using tree point clouds, and the projected tree points formed a series of S curves.

Object classification performance

Through many experiments, we collected 23 LiDAR data in different environments and manually labeled 1056 objects using the developed semi-automatic 3D object labeling tool. The generated 3D object dataset consisted of 335 walls, 223 bushes, 83 pedestrians, and 415 trees. After the proposed CNN model was initialized, 530 object data were input to it as a training dataset and 526 as an evaluation dataset to investigate the 3D object classification accuracy of the proposed method.

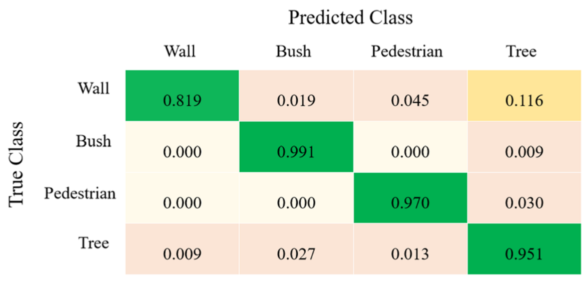
The confusion matrices in Fig. 9 illustrate that the average recognition accuracy for the four object classes was 93.3%. The straightforward structure of bush and pedestrian permitted greater recognition accuracy compared to that of other objects. The misclassification ratio between tree and wall was high, mainly due to the sparse characteristics of LiDAR point clouds and the loss of original information. In addition, when LiDAR was scanning trees at a distance farther away, the obtained tree trunk information was in low density, while the leaves were in high density, which caused the misclassification between walls and trees. To avoid over fitting, the k-fold cross-validation method was applied to our dataset for training and testing. The classification results of bush, pedestrian, tree and wall were 99%, 98.8%, 99.1% and 96.7%, respectively.

Figure 10 shows the object classification results captured in different outdoor scenes. Here, walls, bushes, pedestrians, and trees were accurately classified and coded in blue, red, carmine, and green, respectively.

We also tested our trained model to recognize pedestrian in the Sydney Urban Objects Dataset [31]. The recognition rates of pedestrian reached 90.2%. Thus, our generated 3D object dataset contained almost typical objects, so as to generate an adaptive learning model and have strong compatibility for LiDAR point clouds.

Additionally, the four sample types of pedestrian, tree, pillar and traffic sign in the Sydney Urban Objects Dataset were also used to train and test the proposed CNN model, and the classification accuracy achieved 87.3% on average, as shown in Table 1.

As shown in Table 2, the performance of Hough-based Back Propagation Neural Network (BPNN) and voxel-based CNN [30] algorithms was evaluated on our generated object dataset. The Back Propagation Neural Network (BPNN) had a 10000-neuron input layer, five hidden layers with 2560, 1024, 512, 256, and 64 neurons, respectively, and a 4-neuron output layer. In the voxel-based CNN algorithm, a $32 \times 32 \times 32$ matrix was input into the CNN model which also consisted of two CONV layers, two POOL layers, three FC layers with 2048, 128, 64 neurons, respectively, and the output layer with



True Class \ Predicted Class	Wall	Bush	Pedestrian	Tree
Wall	0.819	0.019	0.045	0.116
Bush	0.000	0.991	0.000	0.009
Pedestrian	0.000	0.000	0.970	0.030
Tree	0.009	0.027	0.013	0.951

Fig. 9 Object classification accuracy for four object classes

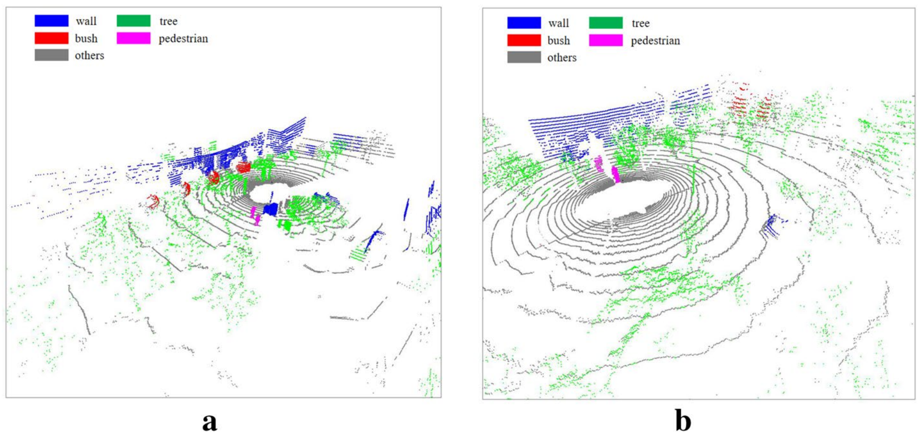


Fig. 10 Object classification results using the proposed method

Table 1 Object classification performance on the Sydney urban objects dataset and our generated object dataset

Object	Our dataset	Sydney urban objects dataset
Pedestrian	0.970	0.980
Tree	0.951	0.857
Pillar	–	0.800
Traffic sign	–	0.850

Table 2 Object classification accuracies for the Hough-CNN, Hough-BPNN, and Voxel-CNN

Types	Hough-CNN	Hough-BPNN	Voxel-CNN
Wall	0.819	0.774	0.655
Bush	0.991	0.954	0.933
Pedestrian	0.970	0.939	0.531
Tree	0.951	0.684	0.708
Average accuracy	0.933	0.838	0.707

four outputs. The average accuracies for the Hough-based BPNN and the voxel-based CNN algorithms were 83.8% and 70.7%, respectively. The proposed Hough-based CNN method had better classification performance and the average accuracy up to 93.3%.

Conclusions

This paper presented a CNN-based 3D object classification using the Hough space computed from the LiDAR points of 3D objects. Firstly, the 3D points were transformed into a Hough space by HT algorithm. Then, a CNN model was trained to classify four types of objects, including walls, bushes, pedestrians, and trees. Experimental results showed that our object classification accuracy achieved 93.3%. The accuracy of bush and pedestrian objects reached 99.1% and 97%, respectively. In this method, the Hough space of LiDAR point clouds was utilized to classify objects so as to largely overcome the unstructured spatial distribution, disordered arrangement and sparse distribution of point clouds. In future, we will enrich the object classes and quantity of our object dataset, so as to train more adaptive learning models for 3D object classification researches.

Acknowledgements
We acknowledge Enago for the linguistic editing and proofreading during the preparation of this manuscript.

Authors' contributions
LZ, YT and WS implemented the experiments and wrote the whole manuscript. SF and AG revised the manuscript. JL surveyed the related works. All authors read and approved the final manuscript.

Funding
This research was supported by the MSIT (Ministry of Science, ICT), Korea, under the High-Potential Individuals Global Training Program (2019-0-01585) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation), National Nature Science Foundation of China (No. 61503005), the Great Wall Scholar Program (CIT&TCD20190304, CIT&TCD20190305), and "Yuyou" Project of North China University of Technology.

Availability of data and materials
The individual data are available in the archives of the laboratory and can be obtained from the corresponding author on request. The Sydney Urban Objects Dataset can be found at http://www.acfr.usyd.edu.au/papers/SydneyUrbanObjectsDataset.shtml

Competing interests
The authors declare that they have no competing interests.

Author details
[1] School of Information Science and Technology, North China University of Technology, Beijing, China. [2] Department of Computer and Information Science, University of Macau, Macau, Taipa, China. [3] Beijing Key Lab On Urban Intelligent Traffic Control Technology, Beijing, China.

Received: 28 September 2019 Accepted: 24 April 2020
Published online: 07 May 2020

References

1. Aamir M, Yi-Fei P, Rahman Ziaur et al (2018) A hybrid proposed framework for object detection and classification. J Inform Processing Syst 14(5):1176–1194
2. Seo YS, Huh JH (2019) Automatic emotion-based music classification for supporting intelligent IoT applications. Electronics 8(2):164
3. Zhang J, Wang W, Lu C, Wang J, Sangaiah AK (2019) Lightweight deep network for traffic sign classification. Annals Telecommun. https://doi.org/10.1007/s12243-019-00731-9
4. Chu PM, Cho S, Park J, Fong S, Cho K (2019) Enhanced ground segmentation method for Lidar point clouds in human-centric autonomous robot systems. Human-centric Comput Inform Sci 9(1):1–4
5. Wu J, Tian Y, Xu H, Yue R, Wang A, Song X (2019) Automatic ground points filtering of roadside LiDAR data using a channel-based filtering algorithm. Optics Laser Technol 1(115):374–383
6. Guo Y, Bennamoun M, Sohel F et al (2014) 3D object recognition in cluttered scenes with local surface features: a survey. IEEE Trans Pattern Anal Mach Intell 36(11):2270–2287
7. Ghrabat MJ, Ma G, Maolood IY, Alresheedi SS, Abduljabbar ZA (2019) An effective image retrieval based on optimized genetic algorithm utilized a novel SVM-based convolutional neural network classifier. Human-centric Comput Inform Sci 9(1):31
8. Hao W, Wang Y (2016) Structure-based object detection from scene point clouds. Neurocomputing 191:148–160
9. Zeng H, Liu Y, Li Siqi et al (2018) Convolutional neural network based multi-feature fusion for non-rigid 3D model retrieval. J Inform Processing Syst 14(1):176–190
10. Zhang J, Chaoquan L, Li X et al (2019) A full convolutional network based on DenseNet for remote sensing scene classification. Mathemat Biosci Eng 16(5):3345–3367
11. Meng R, Rice SG, Wang J, Sun X (2018) A fusion steganographic algorithm based on faster R-CNN. Comput Materials Continua 55(1):1–6
12. Rangel JC, Martínez-Gómez J, Romero-González C, García-Varea I, Cazorla M (2018) Semi-supervised 3D object recognition through CNN labeling. Appl Soft Comput 1(65):603–613
13. Xiao L, Wang R, Dai B et al (2018) Hybrid conditional random field based camera-LIDAR fusion for road detection. Inf Sci 432:543–558
14. Rusu RB, Bradski G, Thibaux R, Hsu J et al (2010) Fast 3D recognition and pose using the viewpoint feature histogram. IEEE/RSJ International Conference on Intelligent. p 2155–2162
15. Wohlkinger W, Vincze M (2011) Ensemble of shape functions for 3D object classification. IEEE International Conference on Robotics & Biomimetics. p 2987–2992
16. Chen T, Dai B, Liu D, Song J et al (2014) Performance of global descriptors for velodyne-based urban object recognition. IEEE Intelligent Vehicles Symposium Proceedings. p 667–673
17. Zhu S, Zhang L, Luo Y et al (2017) Characteristics positioning of facial point cloud based on spin image. Comput Engin Design 8:2209–2212

18. Dong Z, Yang B, Liu Y et al (2017) A novel binary shape context for 3D local surface description. ISPRS J Photogrammetry Remote Sensing 130:431–452
19. Salti S, Tombari F, Di Stefano L (2014) SHOT: unique signatures of histograms for surface and texture description. Comput Vis Image Underst 125:251–264
20. Guo Y, Sohel F, Bennamoun M et al (2015) A novel local surface feature for 3D object recognition under clutter and occlusion. Inf Sci 293:196–213
21. Prakhya SM, Liu B, Lin W (2015) B-SHOT: a binary feature descriptor for fast and efficient key point matching on 3D point clouds. IEEE/RSJ International Conference on Intelligent Robots and Systems. p 1929–1934
22. Serna A, Marcotegui B (2014) Detection, segmentation and classification of 3D urban objects using mathematical morphology and supervised learning. ISPRS J Photogrammetry Remote Sens 93:243–255
23. Wang H, Wang C, Luo H et al (2014) Object detection in terrestrial laser scanning point clouds based on hough forest. IEEE Geosci Remote Sens Lett 11(10):1807–1811
24. Becker C, Häni N, Rosinskaya E et al (2017) Classification of aerial photogrammetric 3D point clouds. Photogrammetric Eng Remote Sens 84(5):287–295
25. Jin L, Yihe Y, Shiqi L et al (2019) Attention-based BiGRU-CNN for Chinese question classification. J Ambient Intell Humanized Comput. https://doi.org/10.1007/s12652-019-01344-9
26. Su H, Maji S, Kalogerakis E et al (2015) Multi-view convolutional neural networks for 3D shape recognition. IEEE International Conference on Computer Vision. p 945–953.
27. Zhi S, Liu Y, Li X et al (2018) Toward real-time 3D object recognition: a lightweight volumetric CNN framework using multitask learning. Comput Graph 71:199–207
28. Qi CR, Su H, Mo K, Guibas LJ (2017) PointNet: deep learning on point sets for 3D classification and segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. p 652–660
29. Li Y, Bu R, Sun M, Wu W, Di X, Chen B (2018) PointCNN: Convolution on X-transformed points. Advances in Neural Information Processing Systems. p 820–830
30. Xu Y, Fan T, Xu M et al (2018) SpiderCNN: Deep learning on point sets with parameterized convolutional filters. Proceedings of the European Conference on Computer Vision. p 87–102
31. De Deuge M, Quadros A (2013) Unsupervised feature learning for classification of outdoor 3D scans. In Australasian Conference on Robotics and Automation. (2).

Publisher's Note