1-2020

Fusion of 3D LIDAR and Camera Data for Object Detection in Autonomous Vehicle Applications

Xiangmo Zhao

Pengpeng Sun

Zhigang Xu

Haigen Min

Hongkai Yu
The University of Texas Rio Grande Valley

Recommended Citation

Zhao, X., Sun, P., Xu, Z., Min, H., & Yu, H. (2020). Fusion of 3D LIDAR and Camera Data for Object Detection in Autonomous Vehicle Applications. Ieee Sensors Journal, 20(9), 4901–4913. https://doi.org/10.1109/JSEN.2020.2966034

Fusion of 3D LIDAR and Camera Data for Object Detection in Autonomous Vehicle Applications

Xiangmo Zhao, Pengpeng Sun, Zhigang Xu, Haigen Min, Hongkai Yu

***Abstract*—It's critical for an autonomous vehicle to acquire accurate and real-time information of the objects in its vicinity, which will fully guarantee the safety of the passengers and vehicle in various environment. 3D LIDAR can directly obtain the position and geometrical structure of the object within its detection range, while vision camera is very suitable for object recognition. Accordingly, this paper presents a novel object detection and identification method fusing the complementary information of two kind of sensors. We first utilize the 3D LIDAR data to generate accurate object-region proposals effectively. Then, these candidates are mapped into the image space where the regions of interest (ROI) of the proposals are selected and input to a convolutional neural network (CNN) for further object recognition. In order to identify all sizes of objects precisely, we combine the features of the last three layers of the CNN to extract multi-scale features of the ROIs. The evaluation results on the KITTI dataset demonstrate that : (1) Unlike sliding windows that produce thousands of candidate object-region proposals, 3D LIDAR provides an average of 86 real candidates per frame and the minimal recall rate is higher than 95%, which greatly lowers the proposals extraction time; (2) The average processing time for each frame of the proposed method is only 66.79ms, which meets the real-time demand of autonomous vehicles; (3) The average identification accuracies of our method for car and pedestrian on the moderate level are 89.04% and 78.18% respectively, which outperform most previous methods.**

***Index Terms*—Autonomous Vehicle, Object detection, Object Identification, 3D LIDAR, CNN, Sensor Fusion**

I. Introduction

AUTONOMOUS vehicles can fundamentally improve the safety and comfort of the driving population while reducing the impact of automobiles on the environment [1]. To develop such a vehicle, the perceptual system is one of the indispensable components allowing the vehicle to understand the driving environment, including the position, orientation and classification of the surrounding obstructions. Therefore, sensors such as LIDAR, cameras, radar, sonar have been widely used in the environment sensing system of autonomous vehicles.

3D LIDAR is one of the most prevalent sensors used in the autonomous vehicle perceptual systems, and it has a wide range of view, with precise depth information, and long-range and night-vision capabilities in target recognition [2-4]. In the object detection task, 3D LIDAR has certain advantages over cameras in acquiring the pose and shape of the detected objects, since laser scans contain spatial coordinates of the point clouds by nature [5]. However, the distribution of 3D LIDAR point clouds become more and more sparse as the distance from the scanning center increases, which brings difficulties for a 3D LIDAR to detect specific objects in the classification step.

Cameras can provide high resolution images for precise classification, and the classification methods have been widely used in recent years with extensive research of deep learning in the field of image recognition. Such methods usually first use an object-proposal generation method to generate box proposals, such as the sliding-window [6], edge box [7], select search [8], or multi-scale combinatorial grouping (MCG) [9], and then use the CNN pipeline [10, 11] to perform object-region based recognition. A common disadvantage of those approaches is the high computational costs associated with generating substantial candidate region proposals. Besides, camera suffers from varying illumination and lacking information of the 3D location, orientation and geometry of the object, resulting in imprecise object-region proposals.

In order to obtain highly accurate object location and classification in driving environments, one possible approach is to take full advantage of the complementary information between 3D LIDAR and cameras. For this purpose, we present a multi-object detection methodology, applying the 3D LIDAR-based object-region proposal generator on the point clouds and combining a state-of-the-art CNN classifier on the camera data. The main contributions of this work are three-fold: (1) we present a real-time multi-object detecting system, which performs long-range and high-precision object detection, and (2) propose a fast and accurate method for generating object-region proposals based on the 3D LIDAR data, while maintaining a higher recall rate, and (3) implement a multi-scale CNN model to detect the tiny objects effectively. We are concerned on the representative objects on the road, such as vehicles, pedestrians and bicycles, and the approach can also be extended to some other traffic elements around the moving autonomous vehicles.

To quickly and accurately generate the object-region proposals from 3D LIDAR point clouds, we first encode the unordered original sparse point clouds into a multi-channel matrix according to the time stamp and vertical orientation of each laser beam, and extract ground points by analyzing the range difference between two adjacent beams. The non-ground points were clustered using an adaptive threshold-based cluster algorithm and the bounding box of the clustering will be calculated. Thus, we can reduce the number of pesudo-targets

Manuscript submitted for review August 23, 2019. This work was supported in part by the 111 Project on Information of Vehicle-Infrastructure Sensing and ITS under Grant No. B14043, and in part by the Fundamental Research Funds for the Central Universities under Grant No. 300102248715 (Corresponding author: P. P. Sun, e-mail: pengpeng.sun@chd.edu.cn).

P.P. Sun and H.G. Min,are with the Traffic Information Engineering & Control Department, Chang'an University, Xian 710064, China.

X.M Zhao, Z.G. Xu are with the Joint Laboratory for Internet of Vehicles, Ministry of Education-China Mobile Communications Corporation, Chang'an University, Xi'an 710064, China.

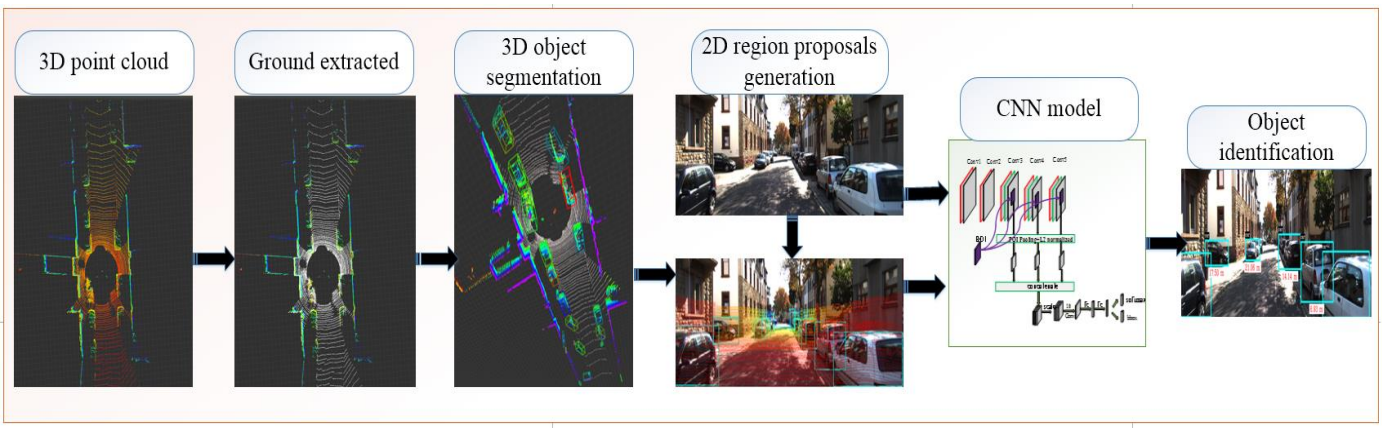


Fig. 1: Overview of the proposed framework for the multi-object detection algorithm.

based on the predefined position and the size of objects. Then, on the basis of the corresponding spatial coordinates between the 3D LIDAR and the camera, the detected bounding boxes were projected back into the image space to create the 2D object-region proposals in the image. In this way, we can narrow the search range in the image and speed up the detection algorithm. Those candidate regions were then processed by a CNN classifier for multi-object recognition. The architecture of the proposed multi-object detection algorithm can be seen in Fig. 1.

The remainder of this article is arranged as follows: The section II surveys the previous related works. The section III depicts the proposed multi-object detection method in detail. The section IV gives the related metrics to evaluate the performance of the proposed method, and discusses the experimental results on the KITTI benchmark dataset [12]. Conclusions are made in section V at last.

II. Related Work

This section gives a concise review of previous works related to 3D-LIDAR-based object detection, camera-based object detection and multiple sensor fusion for object detection.

A. 3D LIDAR Object Detection Approaches

There exists many works on autonomous vehicles covering object detection using 3D LIDAR. Usually, the object detection task based on LIDAR can be divided into two steps: extraction of object region proposals and classification of the objects.

For the sake of extracting the object region proposals, it is usually to encode 3D point clouds which are captured from the 3D LIDAR using a voxel grid [13-16]. Wang et al. [13] encoded the point clouds into 3D feature grid. Then, the 3D detection window slides in the feature grid, and the score of the object is directly voted to the discrete position of the sliding window. An improved approach based on voting strategy can be found in [14]. This work performs object detection in 3D point clouds with a convolutional neural network constructed from sparse convolutional layers based on the voting scheme and it obtains a faster speed. Li et al. [15] extends fully convolutional network (FCN) to 3D and designed a 3D-FCN on voxel grids built on the LIDAR point cloud for vehicle detection. Zhou et al. [16] presented an efficient deep network architecture called VoxelNet for point cloud, which extracts features directly on sparse points of the 3D voxel grid and achieves remarkable results in the KITTI benchmark. One of the advantages of object region proposals based on a voxel representation is that the computational cost is only proportional to the total number of voxels contained in the grid rather than the number of points. However, the precision of the object detection results is slightly reduced due to the fact that the grid size in the map is much lower than the distance accuracy of the 3D LIDAR data. In addition to operating directly on the voxel grid map, some of the previous algorithms first projected the 3D point clouds onto 2D surfaces as the depth map and then used some image-like methods to generate region proposals [5, 17]. Li et al. [5] detects a car by projecting the 3D point clouds into the front view to obtain the depth map, and then applys a fully convolutional network to the map to predict the 3D box of vehicles, and obtained a comparable performance on the KITTI object benchmark dataset [12]. Minemura et al. [17] proposed an improved method called LMNet, which represents the point cloud as five frontal-view maps (i.e., Reflection, Range, Distance, Side, Height) and is used to input LMNet for multiclass detection. However, projecting the 3D point clouds to a 2D view will lose a lot of important information, and this information could be critical for robust detection of objects, especially for detecting objects in crowded scenes. Another method widely used is to divide points into clusters with characteristics. For example, when dealing with the 3D point clouds captured by an autonomous vehicle, simply removing the ground points and aggregating the remaining points can produce a reasonable segmentation [18]. Finer segmentation can be achieved by forming graphics on the point clouds [19, 20]. Recently, PointNet [21], PointNet++ [22] were proposed for processing point sets, and have shown to work reliably well in indoor environments. Such approaches do not need to carry on any kind of mapping transformation of the point clouds, and operates at the point clouds level. Thus, those methods are more versatile and can use various 3D LIDAR sensors.

To classify the object-region proposals, some early studies mainly concentrated upon the hand-crafted features which come from the spatial relation among the LIDAR points or the intensity characteristics of them, e.g., spin image [23], fast point feature histogram (FPFH) [24], and traditional classification techniques, e.g., SVM [25], MLP and Ada Boost [26, 27]. In reference [25], a classifier based on SVM is proposed, which divides the clusters into ground, vegetation, construction and vehicles. A total of 13 features are extracted as the input to the SVM classifier. However, these traditional classifiers have weak generalization ability and low recognition precision, which can't meet the requirement of the recognition accuracy of the perception system of autonomous vehicle. The recently developed deep learning object detection algorithms, such as VeloDeep [14], VoxelNet [16] are more general and robust than the above methods because they can identify more object categories [28]. However, with the increase of the amount of point clouds data involved in computing 3D network model, the computational power and memory requirements for the computation of the 3D network model are increased in cubic terms.

B. Camera-based Object Detection Approaches

Following the conventional learning or feature-based object detection paradigm, deep learning has shown excellent performance in the field of object detection using cameras for intelligent transportation systems (ITS) application. The state-of-the-art methods of object recognition using deep learning can be roughly divided into two categories: the region-based method and the end-to-end method. The general process of a region-based approach is to generate a large number of candidate bounding boxes from the image using common methods like a sliding window [6], a selective search [10], and the features of each object-region box would be extracted and classified by a convolutional neural network model [29-31]. R-CNN [30] is a milestone applying CNN approach to object detection, and it achieves excellent object detection accuracy. On this method, a selective search [10] was used to generate region proposals, and the object image extracted by the proposal was normalized as the standard input of the CNN. However, in classification, it needs to extract features from each extracted proposal of the test image, and the repetitive feature extraction leads to a huge computational waste. He et al. [31] improved the efficiency of R-CNN [30] by accelerating the feature extraction link. In his method, the convolution feature map of the whole input image is calculated, and then the feature vectors extracted from the shared feature map are used to classify each object. This method is like to R-CNN [30], the training process of the network is still isolated, i.e., extracting the candidate regions, calculating CNN features and SVM classification are carried out separately. This method needs to pass a large number of intermediate results in the network besides the overall training parameters. In the fast R-CNN [29], a breakthrough idea was put forward, which combines the classification and bounding box regression. The training process is unified with further integration of the multiple loss layer, which improves the accuracy of the algorithm. Faster R-CNN [32] is the first framework to unify the generation of object candidate region, feature extraction and object classification into a convolutional neural network, which improves the efficiency of the whole object detection system. However, this method is does not achieve good performance on small object detection. To address this issue, Li et al. [33] developed a scale aware Fast R-CNN pipeline, which embeds multiple built-in sub-networks and can detecte pedestrians from a scale that does not intersect.

In the cases of the end-to-end method, object detection is modeled as a regression problem to attempt to discard the links that generated the object-region proposals [34-36]. In YOLO (You Only Look Once) [34, 35], the image is divided into a fixed size of grid, for each grid the object position and the confidence degree will be predicted. The network output layer is mapped to the above results of the grids, thus achieving end-to-end training. The network of Fast YOLO [35] is further simplified, speed up the detection algorithm to 155 frames per second (fps). An improved method for the tiny object detection, namely, SSD (single shot detector) [36] evaluates the candidate object-region and category confidence maps by using different layer features in the convolutional layer and achieves higher detection accuracy. The detection rate of these speeding up methods can reach more than 30 fps. However, the speed of the algorithm comes at the cost of accuracy.

C. 3D-LIDAR and Camera Fusion Approaches

Different sensors have their own merits but there are also some problems. 3D LIDAR is mainly used for 3D measurement and can't be affected by the ambient lighting, but it provides little information about the appearance of objects. In contrast, cameras can provide rich texture information of the detected objects, but their performance greatly depends on illumination conditions. Therefore, multi-sensor information fusion is critical for accurate object detection, but the fusion of sensor information should be based on accurate sensor calibration. Recently, many studies are emerging on multi-sensor data fusion, and a survey can be referenced in [37]. Normally, the fusion techniques can be divided into three categories based on the level of abstraction that occurs, including (1) fusion on the pixel level which combines the measurements to create a new type of data [38], (2) fusion on the feature level that integrates features coming from data from different sensors [39-43] and (3) fusion on the decision level which combines the classified results from the data of each sensor [2, 44]. Schoenberg et al. [38] fused the LIDAR with the camera image on a pixel-level, and for each LIDAR point there is a pixel in the image corresponding to it. Therefore, each point is added a pixel of color intensity information. This method only uses of the intensity information and suffered from non-overlapping region problems. An improved approach presented by Cho et al. [39], who extracted the data features of each sensor respectively, and combines them to classify and track the moving objects. The work in [40] performed a pedestrian detection task by combining the 3D-LIDAR data and the RGB image on different levels of the convolutional nets. The point clouds were first converted into horizontal disparity, height, angle (HHA) maps,

and then the HHA maps and image were passed to two different CNN models for classification. Chen et al. [41] proposed a multi-view network (MV3D) for 3D object detection, which combines multiple views of LIDAR point cloud and images for 3D object proposals and object identification. An improved deep model called AVOD [42] is proposed for small object classes that multi-modally fuses features generated by point clouds and RGB images to generate high-resolution feature maps to generate reliable 3D object proposals. Liang et al. [43] exploits continuous convolutions to fuse image and LIDAR feature maps at different levels of resolution for 3D object detector. Oh et al. [44] proposed an object detection method based on the decision-level fusion, which fused the classification outputs from 3D LIDAR and the image data and obtained a classification performance of 77.72%. Instead of detecting the objects separately from the 3D LIDAR point clouds or the image, it fuses the final results detected by the two sensors. In this paper, we just use the 3D LIDAR data to extract object-region proposals to obtain the object's initial location, and use a CNN network model to extract the feature from the corresponding image region and identify the object in the region. The superiority of our method is to take full advantage of the ability of 3D LIDAR to locate object quickly and accurately, and the merit of image for object recognition.

III. Object Detection System

The framework of the proposed object detection algorithm is shown in Fig. 1. This approach has two modalities of input, including 3D point cloud captured by a Velodyne 64E LIDAR and color images captured by a CCD sensor, which are derived from the KITTI benchmark dataset [12]. The dataset was already calibrated by providing synchronized and calibrated data. The proposed framework is made up of two parts: (1) the generation of object-region proposals, including the pre-processing of 3D LIDAR point clouds, extraction and removal of ground points, clustering non-ground obstacles, calculating the 3D bounding boxes (BBs) of clustered obstacles and projecting the BBs onto an image to generate 2D object-region proposals, and (2) a multi-scale CNN-based classifier used to classify the object-region proposal.

A. Object-Region Proposal Generation Using 3D LIDAR Data

When an autonomous vehicle is moving, it may encounter various sized objects from all directions and locations. To accelerate the detection process, the state-of-the-art approaches generally use a proposal generator to generate a set of candidate regions instead of exhaustive window search. The presented method only utilizes 3D spatial information provided by a 3D LIDAR to generate the object-region proposals, which can be divided into 3 steps as below.

(1) Ground Point Extraction and Removal: In the 3D point cloud captured by 3D LIDAR, all the points that hit the obstacles on the ground, such as cars, trees, vegetation are always connected to the points on the ground. In order to improve the quality of the object-region proposals and to reduce unnecessary computation, we need to remove the ground points from the raw point cloud before performing object clustering. One common method of ground extraction and removal is to discard all points within a certain height [45]. Such method may play well in simple scenarios, but fails when the vehicle is moving in complex road environment. Li et al. [46] introduced an improved method by projecting measurements into a polar grid cell, where if both the mean height and the standard deviation are within the predefined thresholds, the region within the grid cell will be considered to belong to the ground set. However, even with this approach, an off-road environment may still be a challenge, and the operation could also be time consuming. The distance between adjacent rings is more sensitive than the vertical displacement for measurement of the terrain slope [1]. The analysis of the range difference between adjacent rings provides a new idea for reliably detecting obstacles that are not even obvious to the vertical threshold algorithm [47, 48]. Choi et al. [47] compares the radius difference between adjacent beams with the given threshold to identify the ground points. Since the actual radial difference between adjacent beams varies with the attitude of the vehicle, it is very challenging to set an appropriate threshold. Hata et al. [48] identified curb-like points by checking whether the ring distance between beams is within a given interval, which is based on a fixed ring distance on the plane. In this paper, we still identify ground points analyzed the radius distance between adjacent rings, but in different forms. We use the ratio of the actual measured range difference to the estimated range difference between adjacent rings to avoid the inconsistent variations in the range difference of adjacent rings of 3D LIDAR at different positions. In addition, the estimated range difference between adjacent rings is not a fixed value, but varies with the road conditions.

One of the major challenges in processing the 3D LIDAR data is that the 3D point cloud's elements are represented by Cartesian coordinates $p=[p_x, p_y, p_z, p_I]$, which contain a large number of discrete and unordered 3D points of the scenes. It is a time-consuming procedure to execute the search and index operations among the points. Therefore, it is necessary to reorganize the original disordered sparse 3D LIDAR point clouds into the ordered point clouds.

Actually, the raw output data of the 3D LIDAR is based on a spherical coordinate system, which mainly includes the azimuth angle θ, the pitch angle of each beam ϕ, the measurement distance d and the reflect intensity I. Therefore, we can encode the disordered sparse point cloud P into a multi-channel dense matrix M according to the rotation angle of the points and the number of the rings that the points belong to (i.e., the ID of the source laser beam), as illustrated in Fig. 2. The number of rows is defined by the numbers of rings in the 3D LIDAR frame. The number of columns depends on the rotation rate of the Velodyne LIDAR, which is 10 Hz. And for each rotation, the LIDAR sensor generate 64×2048 laser points.

We first aggregated the point cloud P into the cells of matrix $b_{r,c}$ by the similar method from the previous work [5], which can be described through Eq. (1) to Eq. (5).

$$p_\theta = \text{atan2}(p_y, p_x) \quad (1)$$

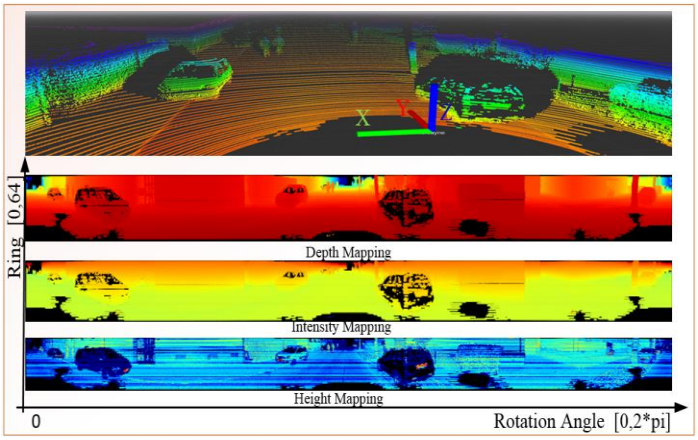


Fig. 2. Example of 3D LIDAR point clouds from the KITTI benchmark dataset followed by the corresponding depth mapping, height mapping and reflectance mapping. Each row represents the measurement of a single laser beam done during one rotation of the sensor. Each column contains measurements for all 64 laser beams captured at a specific rotational angle at the same time.

$$p_\phi = \arcsin(p_z / \sqrt{p_x^2 + p_y^2 + p_z^2}) \quad (2)$$

$$p_r = \lfloor (p_\theta + 180) / \Delta\theta \rfloor \quad (3)$$

$$p_c = \lfloor p_\phi / \Delta\phi \rfloor \quad (4)$$

$$b_{r,c} = \{ p \in P \mid p_r = r \cap p_c = c \} \quad (5)$$

Where $p = [p_x, p_y, p_z, p_I]$ represents to a 3D point, (p_θ, p_ϕ) represents the rotation angle and pitch angle of the point, (p_r, p_c) represents the row and column indices of a point in the matrix, $\Delta\theta$ represents the average rotation angle resolution, and $\Delta\phi$ represents the vertical angle resolution of the continuous beam transmitter. In fact, the row also corresponds to the number of laser beams and all the points that allocated to the same row are captured by the same laser beam.

Since the horizontal representation of our encoding is equal to the original Velodyne resolution, then a few points may fall into the same cell $b_{r,c}$, in which case the point closest to the observer is retained. We reduce the number of channels and populate the cell $b_{r,c}$ with the 3-channel data $m(b_{r,c})$ which can be expressed by Eq. (6)

$$m(b_{r,c}) = (p_z, p_I, p_{depth}) \quad (6)$$

Where p_z, p_I, p_{depth} represent the height, intensity value and depth value of a point, respectively. The depth value is defined in Eq. (7)

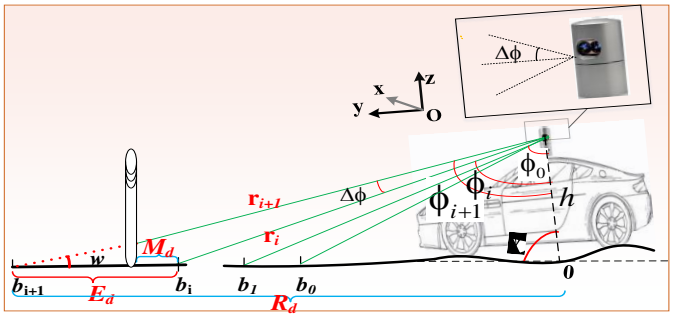


Fig. 3: The geometrical model for ground extraction is established by comparing the expected range difference Ed with the measured range difference md between the two adjacent 3D LIDAR rays on the ground.

$$p_{depth} = \sqrt{p_x^2 + p^2_y} \quad (7)$$

An example of transformation of 3D LIDAR point cloud from the KITTI benchmark in the multi-channel dense matrix is shown in Fig. 2, where each row represents the measurement of a single laser beam done during one rotation of the sensor. Each column contains the measurements of all 64 laser beams captured at a specific rotational angle at the same time. This transformation provides an image-like coordinate frame to organize discrete points and it also keeps the spatial relationship between the points.

On the ideal flat horizontal plane, it is assumed that the height of a 3D LIDAR installation and the pitch angle of each laser beam are known and the expected depth difference between the two adjacent beams can be computed. The difference in this range decreases with the rising elevation of the surface. A geometrical model of ground extraction algorithm is shown in Fig. 3.

Suppose that the symbol $b_{i+1,j}$ is used to represent the cell of $(i+1)$th row and jth column of the matrix, and the symbol $p_{depth}^{i+1,j}$ is used to represent the depth value of points in $b_{i+1,j}$. In order to determine if the points in $b_{i+1,j}$ are ground points, we first estimate the depth difference between the previous cell of the same column (i.e., $b_{i,j}$), and use the symbol $E_d(b_{i,j}, b_{i+1,j})$ to represent the estimated depth difference. The 3D LIDAR points of two adjacent scan lines on the plane will form a concentric circle, and the depth difference between the two adjacent scan lines depends on the installation height of 3D LIDAR and the pitch angle in the vertical direction of the laser line. The $E_d(b_{i,j}, b_{i+1,j})$ value is a constant, and its value depends on the pitch angle of the adjacent (ith and $(i+1)$th) scan lines in the vertical direction and the installation height of the LIDAR. The actual depth difference between the adjacent cells $b_{i,j}$ and $b_{i+1,j}$ is called the measured depth difference, and is represented by $M_d(b_{i,j}, b_{i+1,j})$. The measured depth difference $M_d(b_{i,j}, b_{i+1,j})$ on the ground seldom changes, and the estimated depth difference $E_d(b_{i,j}, b_{i+1,j})$ is approximately equal to the measured value $M_d(b_{i,j}, b_{i+1,j})$. However, when the points of 3D LIDAR in the cell $b_{i+1,j}$ hit the obstacle as shown in Fig. 3, the depth of the points are truncated by the obstacles, resulting in a sudden decrease in the depth distance between the two adjacent points of two adjacent laser line. It wills lead to an obvious difference between the estimated depth difference $E_d(b_{i,j}, b_{i+1,j})$ and the measured depth difference $M_d(b_{i,j}, b_{i+1,j})$. Therefore, we can compare the values of $E_d(b_{i,j}, b_{i+1,j})$ and $M_d(b_{i,j}, b_{i+1,j})$ to determine whether the points in the cell $b_{i+1,j}$ are ground points or obstacle points. The LIDAR point cloud is approximately concentrically distributed on the ground, and the farther the adjacent rings are from the origin of LIDAR, the greater the value of $E_d(b_{i,j}, b_{i+1,j})$. The range of absolute difference between $M_d(b_{i,j}, b_{i+1,j})$ and $E_d(b_{i,j}, b_{i+1,j})$ is $[0, E_d(b_{i,j}, b_{i+1,j})]$, thus this range varies with position, and it is difficult to find a suitable threshold to distinguish the category of LIDAR point cloud, but the proportional range of $M_d(b_{i,j}, b_{i+1,j})$ and $E_d(b_{i,j}, b_{i+1,j})$ at any

position is always [0,1] . Therefore, Instead of using the absolute difference, we adopt a proportional method to avoid the inconsistent variations in the depth difference of two adjacent laser lines of 3D LIDAR at different positions. Accordingly, and the ground attestation of cell $b_{i+1,j}$ can be calculated by Eq.(8).

$$\mathrm{P}(b_{i+1,j}) = \frac{M_d(b_{i,j}, b_{i+1,j})}{E_d(b_{i,j}, b_{i+1,j})} \tag{8}$$

Where $M_d(b_{i,j}, b_{i+1,j})$ is the actual depth difference between adjacent cells $b_{i,j}$ and $b_{i+1,j}$, and $E_d(b_{i,j}, b_{i+1,j})$ is the estimated depth difference between the adjacent cells. The $M_d(b_{i,j}, b_{i+1,j})$ value is calculated with Eq. (9) as below.

$$M_d(b_{i,j}, b_{i+1,j}) = \left| {p_{\text{depth}}}^{i+1,j} - {p_{\text{depth}}}^{i,j} \right| \tag{9}$$

Where ${p_{\text{depth}}}^{i+1,j}$ represents the depth value of points in the cell $b_{i+1,j}$. The geometrical model of the ground extraction algorithm is illustrated in Fig. 3. Due to the variety of terrain, the vehicle may encounter flat, undulating, hillsides or other roadways. The extension line of the LIDAR's axis is perpendicular to the surface flat road, i.e., the angle between the extension line of the LIDAR's axis and the ground surface is 90º. However, for a sloping road, the extension line of the LIDAR's axis is no longer perpendicular to the road surface due to the pitch of the vehicle, as shown in Fig.3. In order to make the proposed algorithm adaptive to different roads, it is not always assumed that the extended line of the LIDAR axis is perpendicular to the ground plane when calculating the expected radial distance between two adjacent scanning lines. Here, the angle between the extension line of the LIDAR's axis and the ground surface is defined as a variable γ , which varies with the pitch angle of the vehicle.

According to the geometrical relation, $E_d(b_{i,j}, b_{i+1,j})$ can be calculated by Eq. (10).

$$\frac{E_d(b_{i,j}, b_{i+1,j})}{\sin\Delta\phi} = \frac{h}{\sin\omega} \tag{10}$$

Where h represents the installation height of 3D LIDAR, $\Delta\phi$ represents the vertical angle resolution of 3D LIDAR, ω represents the angle between the ground surface and the $(i+1)$th scan line, and can be calculated by Eq. (11).

$$\omega = \pi - \phi_{i+1} - \gamma \tag{11}$$

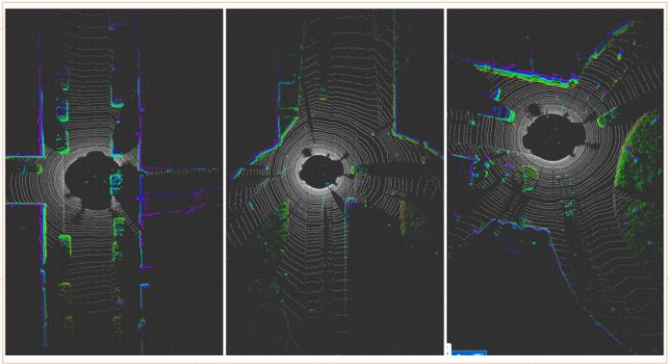


Fig. 4. Examples of 3D LIDAR ground point cloud extractions from the KITTI benchmark dataset, and the white dots indicate the extraction of ground points.

Where ϕ_{i+1} represents the vertical pitch angle of the $(i+1)$th scan line. According to the geometrical relation, γ can be calculated by

$$\frac{r_i}{\sin\gamma} = \frac{R_d}{\sin\phi_i} \tag{12}$$

$$R_d{}^2 = h^2 + r_i^2 - 2hr_i\cos\phi_i \tag{13}$$

Where ϕ_i represents the vertical pitch angle of the ith scan line, and r_i represents the radical distance of the points in the cell $b_{i,j}$. Joint Eq. (3) to Eq. (6), the estimated range difference between two adjacent cells $b_{i+1,j}$ and $b_{i,j}$ can be calculated by Eq. (14).

$$E_d(b_{i,j}, b_{i+1,j}) = \frac{r_i \sin\Delta\phi}{\sin[\arcsin(\frac{h\sin\phi_i}{\sqrt{h^2 + r_i^2 - 2hr_i\cos\phi_i}}) - \phi_{i+1}]} \tag{14}$$

The closer the value $\mathrm{P}(b_{i+1,j})$ is to 1, the greater the probability that the points in the cell $b_{i+1,j}$ belong to the ground set. All the ground cells in the matrix are sequentially extracted by the above method, then we convert those ground cells into point clouds through Eq. (15):

$$\begin{cases} p_x = \dfrac{p_z}{\sin(c\times\Delta\theta)} \times \cos(r\times\Delta\theta) \times \cos(c\times\Delta\phi) \\ p_y = \dfrac{p_z}{\sin(c\times\Delta\theta)} \times \cos(r\times\Delta\theta) \times \sin(c\times\Delta\phi) \\ p_z = p_z \end{cases} \tag{15}$$

After removal of the ground points, we get all the points belong to the obstacle set. Some examples of 3D LIDAR ground point cloud extracted from the KITTI benchmark dataset are shown in Fig. 4, and the white dots indicate the extraction of ground points.

(2) Non-Ground Segmentation: After removing the ground points, the rest of point cloud needs further segmentation. The Euclidean clustering method [49] is one of the most used methods dividing points into individual clusters. This method requires a fixed radius threshold. However, the point cloud captured from the 3D LIDAR is dense horizontally while sparse vertically, which causes the distribution of the points of the object is fairly irregular. Therefore, under a fixed threshold, the segmentation of non-ground points will result in an under-segmentation or over-segmentation problem.

To avoid this problem, the non-ground points are segmented in two steps. We first use a small azimuth difference threshold to cluster the non-ground points into several groups, as illustrated in Fig. 5 (b), and then an adaptive threshold method is used to further segment the clustered groups, as illustrated in Fig. 5 (c).

The segmentation process is described as the following pseudo code. The input is a set of non-ground point clouds P captured from a 3D LIDAR and the output is a set of clusters Γ , in which each cluster contains a set of non-ground points that belong to a single object.

	Algorithm: Segmentation of non-ground points
1	**INPUT:** non-ground points P from 3DLIDAR, the difference azimuth threshold $\theta_{similarity}$
2	**OUTPUT:** object segments $\Gamma = \{C_1, C_2, ..., C_n\}$, set of clusters

Algorithm: Segmentation of non-ground points

INITIALLY: $\Gamma \leftarrow \varnothing$ as the set of clusters to keep
Foreach $p_i \in \mathrm{P}$ **do**
  isInserted $\leftarrow$ false
  Foreach $C \in \Gamma$ **do**
    Foreach $p_j \in C$ **do**
      If $d\_azimuth(\mathrm{p}_i, \mathrm{p}_j) < \theta_{similarity}$ **Then**
        If $d\_\mathrm{position}(\mathrm{p}_i, \mathrm{p}_j) < d(\mathrm{p}_i)$ **Then**
          $C \leftarrow C \cup \{\mathrm{p}_j\}$
          isInserted $\leftarrow$ *true*
          Break;
        End
      End
    End
  End
  If isInserted $\leftarrow$ false **Then**
  $C \leftarrow \{\mathrm{p}_j\}$
  $\Gamma \leftarrow \Gamma \cup \{\mathrm{C}\}$
  End
End

Initially, the first point is categorized to the first group. The 3D LIDAR gives the scanning data in the order of azimuth, thus the azimuth angle of the LIDAR point hitting the same object is continuously distributed. If the difference of the azimuth of the two points is smaller than the threshold, they probably come from the same object. For a point $p_i \in \mathrm{P}, (i \neq 1)$ that is not assigned to any other cluster, we first calculate the azimuth of absolute difference $d\_azimuth(p_i, p_j)$ relative to the other elements $p_j \in C$. If the difference is less than $\theta_{similarity}$, it means that p_i is in the same azimuth zone with cluster C, and then we will further determine whether p_i should be inserted into C by comparing the Euler distance between the two points with the adaptive threshold $d(p_i)$. The value of the threshold

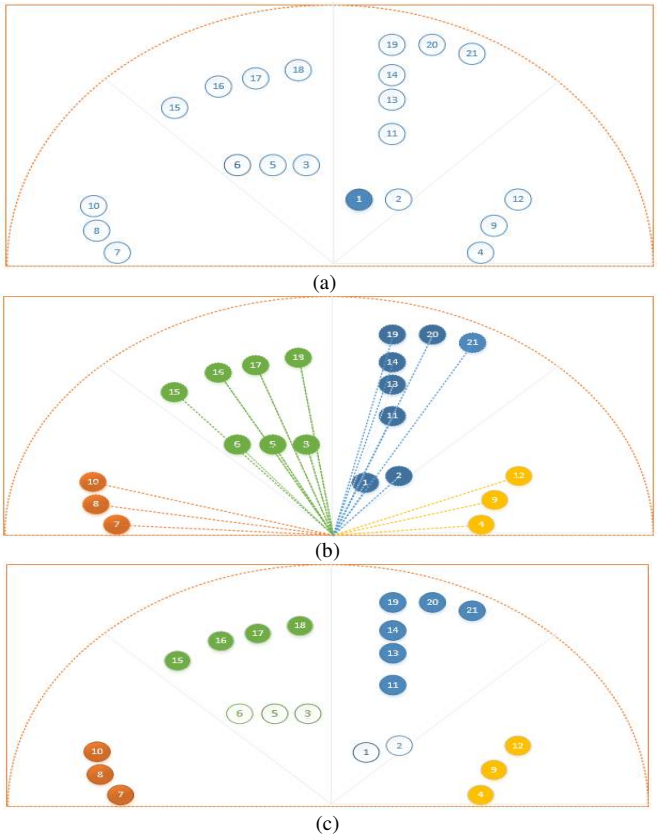


Fig. 5. Illustration of the non-ground segmentation method: (a) shows the original non-ground point clouds; (b) shows the clustering results using the azimuth difference threshold; and (c) shows the final non-ground segmentation results using two criterions.

$\theta_{similarity}$ depends on the horizontal angle resolution of the LIDAR $\Delta\theta$. We take $3\Delta\theta$ as the threshold $\theta_{similarity}$ in order to eliminate the influence of isolated noise points. The function $d\_\mathrm{position}(.)$ is used to calculate the Euler distance between two points. The adaptive threshold $d(p_i)$ is designed as a linear function of the depth values in this point, and can be calculated by:

$$d(p_i) = D_{xy}(p_i) \times u_2 + u_1 \tag{16}$$

The function $D_{xy}(.)$ refers to the depth value between the current point and the origin on the x-y plane. The parameter u_2

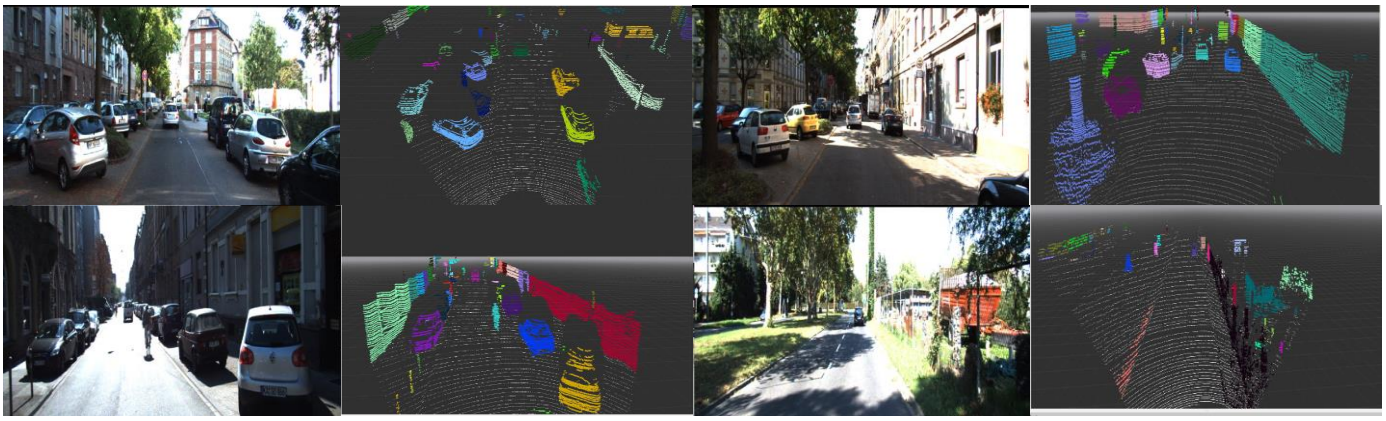


Fig.6 Segmentation results of non-ground point clouds in some typical scenarios, including vehicles in the shade of the trees, darker vehicles, and denser scenes. The proposed algorithm can segment the scene target well.

is obtained by analyzing the regular relationship between two adjacent points in the same laser beam. Considering that the horizontal resolution of the Velodyne HDL-64E is 0.09º when running at 10 Hz, and the interval between two adjacent points in the same laser beam is $0.09\pi \times D_{xy} / 360^{\circ}$ theoretically. As a threshold, the value in this paper is magnified appropriately to triple as parameter u_2. The parameter u_1 serves as the maximum tolerance distance between two obstacles, and this value is also used to distinguish two objects with different horizontal rotation angles, and we use two times the horizontal resolution angle of the 3D LIDAR.

If p_i cannot meet the above conditions, a new cluster is created and p_i assigned to a new cluster. Following the same criteria, it can separate non-ground objects and complete the entire segmentation. An example of a non-ground segmentation results in some typical scenarios is shown in Fig. 6, including vehicles in the shade of the trees, darker vehicles, and denser scenes. The proposed algorithm can segment the scene target well.

(3) Region Proposal Generation: The different processing steps to generate object-region proposals in an image using 3D LIDAR data are shown in Fig. 7.

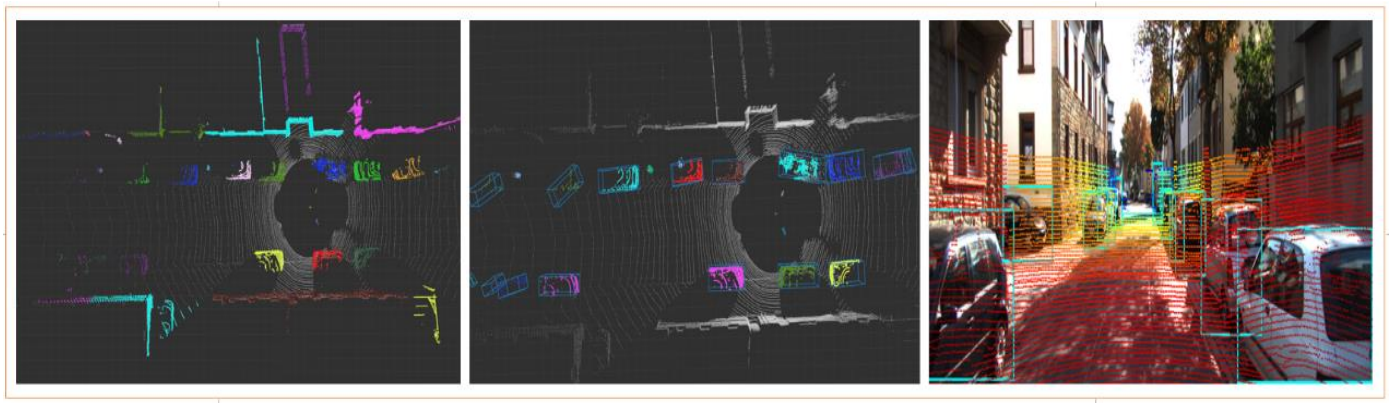
To generate more accurate object-region proposals and ensure better performance of the detector module, we compute the 3D bounding box of each cluster and filter out some dummy objects based on empirical information. When the LIDAR scanning distance exceeds 60 m, few points will be capture. Therefore, we will abandon the candidate box beyond this scope. Besides, the bounding box will be discarded if the width of the bounding box is greater than 3 m, or the length exceeds 10 m, or the height is lower than 0.5 m or greater than 2.5 m. Next, according to the coordinate calibration relationship of the 3D LIDAR and the camera, the remaining 3D boundary boxes are mapped to the corresponding image space. The 3D boundary boxes beyond the image space are discarded, and the 2D candidate boundary rectangle are generated from the mapping area of each 3D boundary box in the image. To guarantee the performance of the detector module, we enlarge the rectangle by 15% so that the entire object is inside the rectangle. The resulting rectangle areas of the image are passed to the CNN model for recognition.

B. CNN-based Feature Extraction and Classification

The CNN model is used to extract the features of the extracted bounding boxes and classify the object in the bounding boxes. The CNN model has achieved remarkable success in the field of object classification due to its ability to learn to express and estimate objects directly. We present a CNN architecture to accurately classify the object-region proposals, as illustrated in the Fig. 8.

The aim is to be able to detect objects that are captured under challenging conditions in which the scale of the object varies dramatically. Although the previous region-based CNN models e.g., Fast-RCNN [29], does not require the proposal box to have a fixed size, but it is difficult to detect the tiny objects robustly. The main reason is that those networks perform ROI pooling only in the last feature map. However, after multiple convolution and pooling operations for the candidate region of a tiny object, there is very little information of the object in the last layer of convolution feature layer. For example, in the VGG-16 model [50], the global strides of 'Conv5' is 16, and when given a bounding box area of less than 16×16 pixel size, the feature of the final output is just one pixel. Under these circumstances, even though the candidate area contains an object, it is difficult to locate and identify the object according to this feature.

To address this issue, the CNN model proposed in this paper does not carry out ROI pooling just on the last convolution feature map. Instead, the region proposal is projected into multiple layers of feature maps, and the ROI pooling operation is executed in each layer. More specifically, our model is based on the VGG16 [50]. Rather than performing ROI pooling only on the last convolutional layer, we execute ROI pooling after Conv3, Conv4 and Conv5 layers. Each layer will generate a fixed-size feature tensor. In order to bring the feature maps from different convolution layers to the same scale, we normalize the feature tensor using L2 normalization for robustness of the detection system, and concatenate all the normalized feature tensors similar to [51]. The normalization is conducted within each pixel of the feature maps, and all the feature maps are treated independently the normalization procedure is expressed with Eq. (17) and Eq. (18) as below.



(a) (b) (c)

Fig. 7. Illustration of the object-region proposals generated in an image using 3D LIDAR data: (a) the results of non-ground clustering; (b) the rest of the 3D bounding boxes after filtering with the experimental information; (c) the projection of the 3D bounding boxes in the 2D image space and obtained the final 2D object-region proposals in the image space after enlarging the 2D bounding boxes

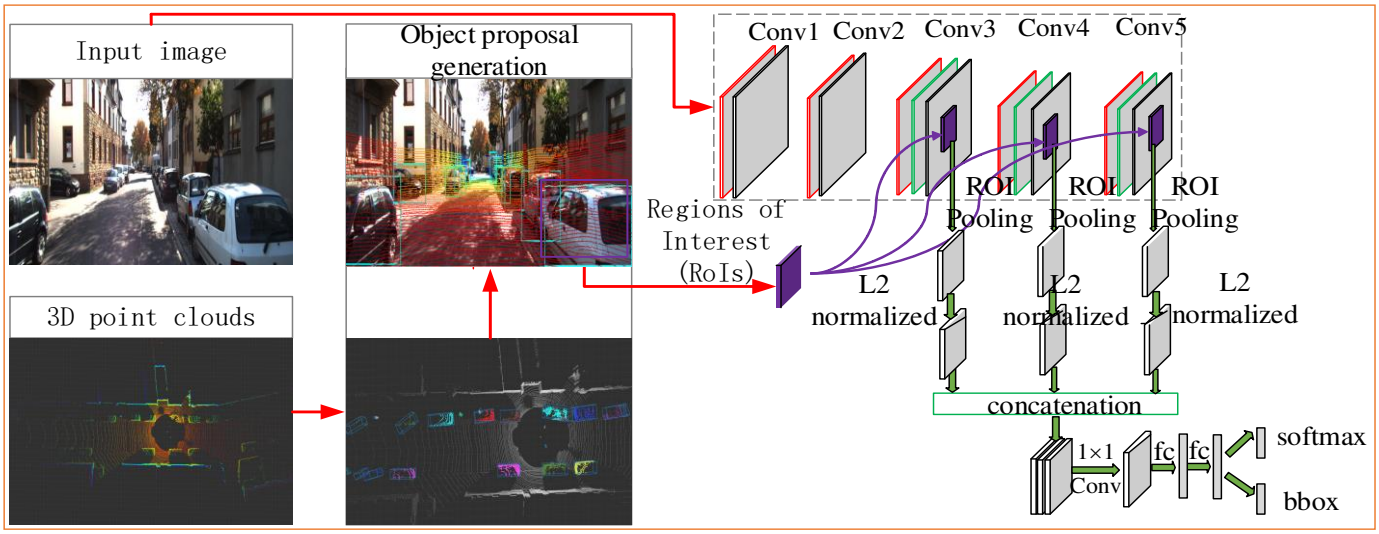


Fig. 8. Structure of the convolutional neural network. The image and the acquired 2D candidate regions are used as input to the proposed network model. The architecture is based on the VGG16 model [50], which consists of five sets of convolution layers: Conv1 to Conv5. We add ROI pooling layers and L2 normalization after Conv3-Conv5 layers to get multi-scale information. Then a 1×1 convolution is used to integrate the information and dimension reduction of the concatenated features. Then we estimate the bounding boxes and class confidence by following two fully connected layers and multitask function.

$$\mathrm{x} = \frac{\mathrm{x}}{\|\mathrm{x}\|_2} \tag{17}$$

$$\|\mathrm{x}\|_2 = (\sum_{i=1}^{d} |x_i|)^{1/2} \tag{18}$$

Where x represents the original features and x represents the normalized features. In Eq. (17), d represents the dimension of the feature from each convolution layer. In the training process, the feature normalization step will redress the scale factor using the updated scale factors. For each channel of the feature map, the scale factor is calculated by Eq. (19).

$$\mathrm{y_i} = \lambda_\mathrm{i}\, \mathrm{x_i} \tag{19}$$

Where $\mathrm{y_i}$ represents the re-scaled feature value. According to the back-propagation rule, the scale factor λ_i can be renovated by Eq. (19) to Eq. (22).

$$\frac{dl}{d\hat{\mathrm{x}}} = \lambda \frac{dl}{dy} \tag{20}$$

$$\frac{dl}{d\mathrm{x}} = \frac{dl}{d\hat{\mathrm{x}}}\left(\frac{\mathrm{I}}{\|\mathrm{x}\|_2} - \frac{\mathrm{x}\mathrm{x}^T}{\|\mathrm{x}\|_2^3}\right) \tag{21}$$

$$\frac{dl}{d\lambda_i} = \sum_{y_i} \frac{dl}{y_i} x_i \tag{22}$$

Where $y = [\mathrm{y}_1, \mathrm{y}_2, \cdots, y_d]^T$.

To match the original size of the ROI pooling feature map, we use 1×1 convolution to narrow the connected feature dimensions. The final feature tensor is then passed to the two fully connected layers for object positioning and recognition based on the feature tensor.

The output of the network model consists of two parts. One is a vector of K+1 dimension output by One-hot encoding, denoted as $\mathrm{p} = \{\mathrm{p}_0, \mathrm{p}_1, \mathrm{p}_2, \ldots \mathrm{p}_K\}$, which represents the probability distribution of which category a sample belongs to. Other outputs a vector representing 4 parameterized coordinates, denoted as $\mathrm{b} = \{\mathrm{b}_{cx}, \mathrm{b}_{cy}, \mathrm{b}_w, \mathrm{b}_h\}$, which represents predicted bounding box location for each of the K object classes. $\mathrm{b}_{cx}, \mathrm{b}_{cy}, \mathrm{b}_w$ and b_h denote the two coordinates of the predicted bounding box center, width and height respectively. For instance, we assume that the ground-truth class label distribution is denoted as a vector $\mathrm{q} = \{\mathrm{q}_0, \mathrm{q}_1, \ldots \mathrm{q}_i, \ldots \mathrm{q}_K\}$, where $\mathrm{q}_{i-1} = 1$ when the sample belong to category i, and the other elements of the vector are 0. We assume the location of the ground-truth bounding-box location is $g = \{\mathrm{g}_{cx}, \mathrm{g}_{cy}, \mathrm{g}_w, \mathrm{g}_h\}$. For object classification and bounding box regression, we defined the multi-task loss (classification loss and bounding box regression loss) function on the ROI during the training phase following [29] as Eq. (23):

$$\mathrm{L}(\mathrm{p}, \mathrm{q}, \mathrm{b}, \mathrm{g}) = \mathrm{L}_{cls}(\mathrm{p}, \mathrm{q}) + [\mathrm{q} \notin \mathrm{bg}]\mathrm{L}_{loc}(\mathrm{b}, \mathrm{g}) \tag{23}$$

The classification loss $\mathrm{L}_{cls}(\mathrm{p}, \mathrm{q})$ is cross entropy loss, and calculated as follows:

$$L_{cls} = -\sum_{i=1}^{N}\sum_{j=0}^{K} \mathrm{q}_{i,j} \log(\mathrm{p}_{i,j}) \tag{24}$$

Where N is the number of samples, K is the number of categories, $\mathrm{p}_{i,j}$ is the probability that the model predicts sample i belong to the category j, and $\mathrm{q}_{i,j}$ is the probability that the sample i belong to category j. For the bounding box regression loss $\mathrm{L}_{loc}(\mathrm{b}, \mathrm{g})$ as Eq. (24), we use a Smooth L1 loss between the predicated bounding box location and the ground-truth bounding box location defined in [29]. When q represents the background ROIs, we ignore $L_{loc}(\mathrm{b}, \mathrm{g})$, i.e., $\mathrm{q} \notin \mathrm{bg}$.

$$L_{loc}(\mathrm{b}, \mathrm{g}) = \sum_{i}^{N} \sum_{j \in \{\mathrm{x}, \mathrm{y}, \mathrm{w}, \mathrm{h}\}} \mathrm{smooth}_{L1}(\mathrm{b}_j^i - g_j^i) \tag{25}$$

IV. Experimental Setup and Evaluation

This section first introduces the object detection benchmark and evaluation metrics. Then the experiments are carried out

and experimental results are analyzed and discussed. All experiments were conducted using an Intel (R) Core (TM) i7-4790 3.6 GHz processor, with 64 GB RAM. The graphics card for convolutional network training and testing is a Titan X with 12 GB of memory. The CNN model was implemented using C++ on the Ubuntu 14.04+ROS operating system and trained on the Caffe platform [52].

4.1. KITTI Object Detection Dataset

In order to evaluate the performance of the proposed multi-object detection algorithm, quantitative and qualitative experiments were conducted on the 2012 2D KITTI object detection benchmark [12]. The dataset consists of a synchronized stereo camera image and a 3D LIDAR frame captured from an autonomous vehicle. The camera image is cropped to pixels and rectified to pixels. Specifically, the 3D LIDAR frames are captured from HDL-64E with 64 scanning lines, and can perform 360 scans. If it rotates at a 10 Hz frequency, it can generate 1 million points per second.

The dataset provides 7,481 frames of training and 7,518 frames of testing. Since the labels in the test set were not disclosed, we adhered to [14], and divided the training data into a training set (80%) and a validation set (20%). The training data contains 9 different categories of 51,867 labels: 'car', 'pedestrian', cyclist', 'van', 'truck', 'sitting person', 'tram', 'miscellaneous' and 'don't care' and show road scene of various appearances. In addition, based on the size of the 2D bounding box in the image space and the occlusion conditions, the object samples in the KITTI benchmark are divided into three difficulty levels: easy, moderate and hard.

4.2. Evaluation

Firstly, the performance of the object-region generation method based on 2D recall of the ground truth annotation is evaluated. We used the provided calibration file to project the proposed object onto the 2D image plane and discarded any detections outside the image. The intersection-over-union (IOU) metric is used as the evaluation criterion to evaluate object-region extraction at three different levels of difficulty.

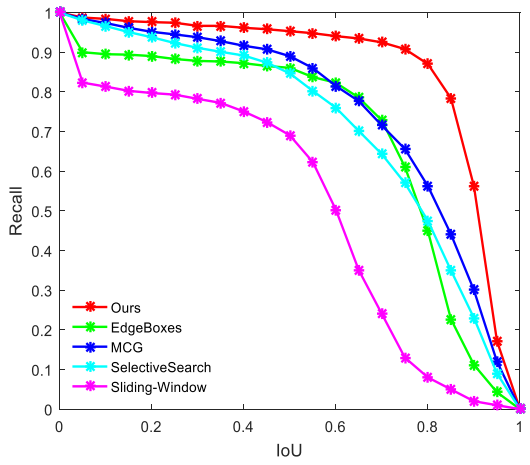


Fig. 9. Recall versus IOU threshold obtained by our proposed region proposal method and other baselines, i.e., sliding window [6], selective search [8], MCG [9], edge boxes [7] on the KITTI validation set.

We evaluated the proposed approach for all 9 object classes in the KITTI validation dataset [12]. We compared our proposed method with other conventional ones such as sliding window [6], edge box [7], selective search [8] and MCG [9], and the detection results are limited to 60 m. The comparison of the recall rates of all methods in generating different object-regions is shown in Fig. 9.

We used 1000 object-region proposals to plot the recall rate as a function of the IOU threshold. As observed, the proposed method provides over 95% of recall rate across the entire range of IOUs.

The main reason is that all baseline methods generate object-region proposals from 2D image space, while the object-region overlap often appears in the image space, and it is difficult to distinguish them. However, in the 3D point cloud captured from the 3D LIDAR, the object-regions can be distinguished by the object depth feature, which is not easy to distinguish in the image space. In addition, the region proposal framework based on visual information can only provide a rough bounding box position. Thus, the recall rate declines rapidly when the higher overlap is required, while the 3D LIDAR has obvious advantage over the camera on achieving the posture and shape of the detected objects, since the laser scans contain the spatial coordinates of the point clouds by nature.

TABLE1
THE RESULTS OF RUNTIME (MS) AND AVERAGE PRECISION (AP%) ON THE KITTI DATA SET IN OUR STUDY COMPARED WITH FOUR STATE-OF-THE-ART PROPOSAL GENERATION METHODS.

Method	NF	Runtime/ ms	AP/%	
			Cars	Pedestrian
Sliding window [6] +Fast-RCN	2000	524	58.8	42.5
Selective search[8] +Fast-RCN	2000	221	73.7	55.9
MCG [9] +Fast-RCN	2000	350	81.3	62.2
Edge boxes [7] +Fast-RCN	2000	139	78.3	62.4
Our method +Fast-RCN	**86**	**53**	**87.8**	**70.7**

To evaluate the performance of the proposed method based on the use of object-region generation method, we used the Fast R-CNN [29] architecture to learn the feature of image, and compared the average precision (AP) and test time with the state-of-the-art object-region generation methods. The network was pre-trained on the PASCAL VOC [53] dataset, and we fine-tuned it for object detection on the KITTI training set and tested it on the KITTI validation set [12]. In the training phase, only three categories, i.e., cars, pedestrians and background were trained for simple experiments. We follow KITTI's assessment method, and use intersection-over-union as an object detection criterion. A detection is accepted if its bounding box in the image space has at least a 50% overlap with the ground-truth. We use the PASCAL VOC [53] evaluation tool kit to calculate the average accuracy. Table 1 compares the accuracy and the calculation time of our study with the existing state-of-the-art studies.

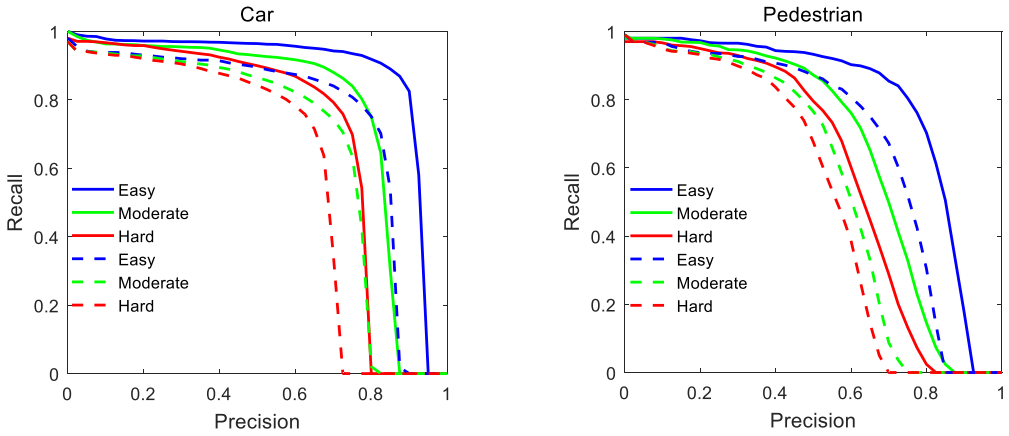


Fig. 10. Precision-recall curves for the three object classes evaluated at three difficulty levels using the KITTI validation set. All precision-recall curves were obtained by our CNN model (solid-line) and VGG16 model (dashed-line).

As can be seen from TABLE 1, our object-region proposal method generates on average 86 non-duplicated proposals per frame (NF), which is smaller than other methods (2000 NF). However, due to our method of providing fewer errors and higher recall rates, we achieved approximately 87% of AP for the cars category achieving better performance than most of the state-of-the-art methods. At the same time, we outperformed the other methods in each category of moderate level by 89.8% and 70.7% for cars, pedestrians respectively, while greatly reduced the calculation time. This clearly shows that the point cloud of 3D LIDAR can be applied to precisely extract object regions at the object level.

To verify the quality of the proposed CNN model, we used the generated region proposal as input and set the original VGG16 [50] model as the baseline. In the experiment, the proposed CNN model was trained on the KITTI benchmark [12] training set, and the employment categories consisting of cars, pedestrians and backgrounds. In the training phase, we first initialized the parameters using a pre-trained VGG-16 with the Image Net [54], and then fine-tuned them using the ground -truth annotations and the generated candidate regions obtained from the KITTI benchmark training set. A sampled candidate region is considered as positive if and only if the candidate region overlaps the ground truth annotation by more than 50%. Otherwise, the candidate region will be treated as a background. The positive samples are a quarter of the total samples. The Nesterov Accelerated Gradient (NAG) [55] algorithm is used for the optimization of the CNN training. NAG is one of the most popular algorithms to optimize neural networks. This method is adaptively updated according to the slope of the loss function in each learning process to accelerate the convergence. We use a NAG optimizer to fine-tune the CNN model, with an initial learning rate of 0.001, a batch size of 16 and a momentum coefficient of 0.9. In addition, instead of fine-tuning all the layers in the experiments, we keep the parameters of the first two sets of the convolution layer unchanged and fine-tune the other layers with maximum number of iterations of 200, 000. After training, we tested the object detection performance of our model's and the baseline approach on the KITTI validation sets using the standard precision-recall (PR) curve. We followed KITTI's assessment method and applied the PASCAL VOC [53] evaluation tool kit to calculate the average precision. Fig.10 shows the precision-recall curve of the baseline method and our method. The area below the precision-recall curve is the AP value. By comparing the precision-recall curves, we can clearly see that our approach greatly exceeds the baseline approach for each grade of difficulty in the three object categories and still performs better with increasing difficulty. This result demonstrates that the information loss can be reduced by combining multiple convolutional feature layers. The results show that by combining the features of multiple convolution layers, the drop of information can be effective decreased and the tiny objects can be detected more effectively, and we have achieved 89.04% and 78.18% of the AP in moderate level for cars and pedestrians respectively, which is superior to most of the published object detection methods. This is the concrete evidence to prove that the proposed method has achieved very competitive results against state-of-the-art methods. Fig. 12 shows some examples of detection in the KITTI dataset. Although there are some serious obstructions and small size objects in the image, the proposed detection method can still be accurately detected. At the same time, we also get the distance information of the target. In order to evaluate the runtime of our proposed approach, we performed a total of 7481 frames of KITTI training and validation datasets. Fig. 11 shows the runtime results of the proposed approach in the experiment. From Fig. 11 it can be seen that the average period is approximately 66.79 ms, which means that our multi-object detection pipeline has a faster frame rate than the 3D LIDAR

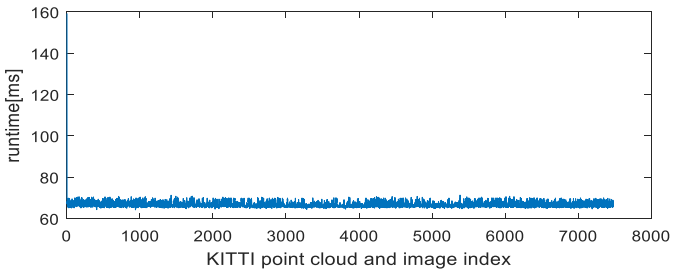


Fig. 11. Runtime for the proposed approach on the KITTI training and validation datasets [12], the average running time of our algorithm is nearly 66.79 ms, which is much lower than the TuSimple's [56] running time.

Fig. 11. Examples of object detection results using our proposed method on the KITTI benchmark dataset [12], including Pedestrians and Cars at various difficulty levels.

frame rate. This illustrates that our approach can be executed rapidly and online.

V. Conclusion and Future Work

In this work, we proposed a novel and fast multi-object detection approach that fully utilizes the complementarity of the 3D LIDAR and camera data to robustly identify multiple objects around an autonomous vehicle. The experimental results of the KITTI benchmark show that this method yields an average of 86 non-repeating object candidate regions per frame, which generates fairly fewer pseudo candidate regions than other conventional methods. In the case of obtaining the object distance information, the average accuracy rates of the proposed method reached 89.04% and 78.18% respectively when detecting the vehicles and pedestrians on moderate difficulty level, which is better than most published methods. The average runtime per frame of our method is about 66.79 ms, meaning that it can be executed rapidly and implemented online. The performance of this method is very competitive comparing to current popular methods.

Although the 3D LIDAR can avoid the effects of environmental illumination changes, few points will be captured when LIDAR scanning range exceeds 60 meters. This will bring difficulties to generate accurate and complete object-region proposals. The limitation of 3D LIDAR scan range will lead to a decrease in performance of the proposed method when detecting tiny objects on moderate or hard levels. To address this problem, in the future, we will use millimeter-wave radar to supplement more information to generate enough object-region proposals. Another limitation of the proposed method is that the method only outputs the 2D bounding box of the object. We will make full use of complementary information to export full 3D bounding boxes

of objects in the future. In addition, the detection of objects in the non-overlapping regions of sensors will also be our focus.

ACKNOWLEDGMENTS

This research was supported in part by the 111 Project on Information of Vehicle-Infrastructure Sensing and ITS under Grant No. B14043, and in part by the Fundamental Research Funds for the Central Universities under Grant No. 300102248715, and the Joint Laboratory for Internet of Vehicles, Ministry of Education-China Mobile Communications Corporation.

Author Contributions: Xiangmo Zhao, Pengpeng Sun correspond for the study and experiments, drafted the manuscript and guided experiments, Zhigang Xu carried out the experiments and edited the manuscript. The other co-authors contributed analysis, discussion, manuscript editing and help with performing the experiments. Author Contributions
Data curation, Haigen Min; Methodology, Pengpeng Sun; Supervision, Xiangmo Zhao; Validation, RunMin Wang; Writing – original draft, Pengpeng Sun; Writing – review & editing, Zhigang Xu.

REFERENCES

[1] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, *et al.*, "Junior: The stanford entry in the urban challenge," *Journal of field Robotics,* vol. 25, pp. 569-597, 2008.

[2] A. Asvadi, L. Garrote, C. Premebida, P. Peixoto, and U. J. Nunes, "Multimodal vehicle detection: fusing 3D-LIDAR and color camera data," *Pattern Recognition Letters,* 2017.

[3] Y. Zhang, J. Wang, X. Wang, and J. M. Dolan, "Road-segmentation-based curb detection method for self-driving via a 3D-LiDAR sensor," *IEEE Transactions on Intelligent Transportation Systems,* 2018.

[4] K. Li, X. Wang, Y. Xu, and J. Wang, "Density Enhancement-Based Long-Range Pedestrian Detection Using 3-D Range Data," *IEEE Trans. Intelligent Transportation Systems,* vol. 17, pp. 1368-1380, 2016.

[5] B. Li, T. Zhang, and T. Xia, "Vehicle detection from 3d lidar using fully convolutional network," *arXiv preprint arXiv:1608.07916,* 2016.

[6] P. Dollár, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 36, pp. 1532-1545, 2014.

[7] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *European conference on computer vision*, 2014, pp. 391-405.

[8] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision,* vol. 104, pp. 154-171, 2013.

[9] J. Pont-Tuset, P. Arbelaez, J. T. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping for image segmentation and object proposal generation," *IEEE transactions on pattern analysis and machine intelligence,* vol. 39, pp. 128-140, 2017.

[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE transactions on pattern analysis and machine intelligence,* vol. 38, pp. 142-158, 2016.

[11] X. Chen, K. Kundu, Y. Zhu, H. Ma, S. Fidler, and R. Urtasun, "3d object proposals using stereo imagery for accurate object class detection," *IEEE transactions on pattern analysis and machine intelligence,* vol. 40, pp. 1259-1272, 2018.

[12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research,* vol. 32, pp. 1231-1237, 2013.

[13] D. Z. Wang and I. Posner, "Voting for Voting in Online Point Cloud Object Detection," in *Robotics: Science and Systems*, 2015.

[14] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner, "Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, 2017, pp. 1355-1361.

[15] B. Li, "3d fully convolutional network for vehicle detection in point cloud," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, 2017, pp. 1513-1518.

[16] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," *arXiv preprint arXiv:1711.06396,* 2017.

[17] K. Minemura, H. Liau, A. Monrroy, and S. Kato, "LMNet: Real-time Multiclass Object Detection on CPU using 3D LiDARs," *arXiv preprint arXiv:1805.04902,* 2018.

[18] H. Wang, B. Wang, B. Liu, X. Meng, and G. Yang, "Pedestrian recognition and tracking using 3D LiDAR for autonomous vehicle," *Robotics and Autonomous Systems,* vol. 88, pp. 71-78, 2017.

[19] H. Yin, X. Yang, and C. He, "Spherical coordinates based methods of ground extraction and objects segmentation using 3-D LiDAR sensor," *IEEE Intelligent Transportation Systems Magazine,* vol. 8, pp. 61-68, 2016.

[20] I. Bogoslavskyi and C. Stachniss, "Efficient online segmentation for sparse 3d laser scans," *PFG–Journal of Photogrammetry, Remote Sensing and Geoinformation Science,* vol. 85, pp. 41-52, 2017.

[21] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE,* vol. 1, p. 4, 2017.

[22] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, 2017, pp. 5099-5108.

[23] M. S. Foster, J. R. Schott, and D. W. Messinger, "Spin-image target detection algorithm applied to low density 3D point clouds," *Journal of Applied Remote Sensing,* vol. 2, p. 023539, 2008.

[24] Y. Tao and J. Zhou, "Automatic apple recognition based on the fusion of color and 3D feature for robotic fruit picking," *Computers and Electronics in Agriculture,* vol. 142, pp. 388-396, 2017.

[25] J. Zhang, X. Lin, and X. Ning, "SVM-based classification of segmented airborne LiDAR point clouds in urban areas," *Remote Sensing,* vol. 5, pp. 3749-3775, 2013.

[26] J. Niemeyer, F. Rottensteiner, and U. Soergel, "Contextual classification of lidar data and building object detection in urban areas," *ISPRS journal of photogrammetry and remote sensing,* vol. 87, pp. 152-165, 2014.

[27] R. O. Chavez-Garcia and O. Aycard, "Multiple sensor fusion and classification for moving object detection and tracking," *IEEE Transactions on Intelligent Transportation Systems,* vol. 17, pp. 525-534, 2016.

[28] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view cnns for object classification on 3d data," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5648-5656.

[29] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440-1448.

[30] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580-587.

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *European conference on computer vision*, 2014, pp. 346-361.

[32] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91-99.

[33] J. Li, X. Liang, S. Shen, T. Xu, J. Feng, and S. Yan, "Scale-aware fast R-CNN for pedestrian detection," *IEEE Transactions on Multimedia,* vol. 20, pp. 985-996, 2018.

[34] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779-788.

[35] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6517-6525.
[36] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, *et al.*, "Ssd: Single shot multibox detector," in *European conference on computer vision*, 2016, pp. 21-37.
[37] F. Garcia, D. Martin, A. De La Escalera, and J. M. Armingol, "Sensor fusion methodology for vehicle detection," *IEEE Intelligent Transportation Systems Magazine,* vol. 9, pp. 123-133, 2017.
[38] J. R. Schoenberg, A. Nathan, and M. Campbell, "Segmentation of dense range information in complex urban scenes," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, 2010, pp. 2033-2038.
[39] H. Cho, Y.-W. Seo, B. V. Kumar, and R. R. Rajkumar, "A multi-sensor fusion system for moving object detection and tracking in urban driving environments," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, 2014, pp. 1836-1843.
[40] J. Schlosser, C. K. Chow, and Z. Kira, "Fusing lidar and images for pedestrian detection using convolutional neural networks," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, 2016, pp. 2198-2205.
[41] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *IEEE CVPR*, 2017, p. 3.
[42] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. Waslander, "Joint 3d proposal generation and object detection from view aggregation," *arXiv preprint arXiv:1712.02294,* 2017.
[43] M. Liang, B. Yang, S. Wang, and R. Urtasun, "Deep Continuous Fusion for Multi-Sensor 3D Object Detection," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 641-656.
[44] S.-I. Oh and H.-B. Kang, "Object detection and classification by decision-level fusion for intelligent vehicle systems," *Sensors,* vol. 17, p. 207, 2017.
[45] C. Mertz, L. E. Navarro-Serment, R. MacLachlan, P. Rybski, A. Steinfeld, A. Suppé, *et al.*, "Moving object detection with laser scanners," *Journal of Field Robotics,* vol. 30, pp. 17-43, 2013.
[46] Q. Li, L. Zhang, Q. Mao, Q. Zou, P. Zhang, S. Feng, *et al.*, "Motion field estimation for a dynamic scene using a 3D LiDAR," *Sensors,* vol. 14, pp. 16672-16691, 2014.
[47] A. Petrovskaya and S. Thrun, "Model based vehicle detection and tracking for autonomous urban driving," *Autonomous Robots,* vol. 26, pp. 123-139, 2009.
[48] A. Y. Hata, F. S. Osorio, and D. F. Wolf, "Robust curb detection and vehicle localization in urban environments," in *Intelligent vehicles symposium proceedings, 2014 IEEE*, 2014, pp. 1257-1262.
[49] Y. Zhou, D. Wang, X. Xie, Y. Ren, G. Li, Y. Deng, *et al.*, "A fast and accurate segmentation method for ordered LiDAR point cloud of large-scale scenes," *IEEE Geoscience and Remote Sensing Letters,* vol. 11, pp. 1981-1985, 2014.
[50] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556,* 2014.
[51] C. Zhu, Y. Zheng, K. Luu, and M. Savvides, "CMS-RCNN: contextual multi-scale region-based CNN for unconstrained face detection," *arXiv preprint arXiv:1606.05413,* 2016.
[52] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 675-678.
[53] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision,* vol. 88, pp. 303-338, 2010.
[54] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 2009, pp. 248-255.
[55] Y. Nesterov, "A method for unconstrained convex minimization problem with the rate of convergence O (1/k^ 2)," in *Doklady AN USSR*, 1983, pp. 543-547.
[56] F. Yang, W. Choi, and Y. Lin, "Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2129-2137.

Xiangmo Zhao, IEEE Member**,** received the B.S. degree from Chongqing University, China, in 1987, and the M.S. and Ph.D. from Chang'an University, China, in 2002 and 2005, respectively. He is currently a professor and the vice president of Chang'an University, China. He has authored or co-authored over 130 publications and received many technical awards for his contribution to the research and development of intelligent transportation systems. His research interests include intelligent transportation systems, distributed computer networks, wireless communications and signal processing.

Pengpeng Sun, received the bachelor's degree in computer science and technology from Chang'an University, Xi'an, China, in 2014, where he is currently pursuing the Ph.D. in traffic information engineering & control. His current research interests include 3D LIDAR point cloud data processing, object detection based on mulit-sensor fusion for autonomous vehicles, computational intelligence and image understanding.

Zhigang Xu, IEEE Member, received the B.S. degree in automation, and the M.S. and Ph.D. in traffic information engineering and control from Chang'an University, China, in 2002, 2005, and 2012, respectively. He is currently an associate professor at Chang'an University, China. His research focuses on connected and automated vehicle, Intelligent Transportation Systems and nondestructive testing of infrastructures.

Haigen Min, IEEE Student Member, received the B.S. and M.S. degrees in the Department of computer science and he is currently pursuing the Ph.D. degree in the Department of Traffic Information Engineering & Control from Chang'an University, China. His research interests include localization and navigation system for intelligent vehicle and test methodology for intelligent & connected vehicle.

Hongkai Yu received the Ph.D. degree in computer science and engineering from University of South Carolina, Columbia, SC, USA in 2018. He then joins the Department of Computer Science at University of Texas-Rio Grande Valley, Edinburg, TX, USA as an assistant professor. His research interests include computer vision, machine learning, deep learning and intelligent transportation system. He is a member of the IEEE.