Cooperative Perception With V2V Communication for Autonomous Vehicles

Hieu Ngo, Hua Fang, and Honggang Wang, *Fellow, IEEE*

***Abstract*—Occlusion is a critical problem in the Autonomous Driving System. Solving this problem requires robust collaboration among autonomous vehicles traveling on the same roads. However, transferring the entirety of raw sensors' data among autonomous vehicles is expensive and can cause a delay in communication. This paper proposes a method called Realtime Collaborative Vehicular Communication based on Bird's-Eye-View (BEV) map. The BEV map holds the accurate depth information from the point cloud image while its 2D representation enables the method to use a novel and well-trained image-based backbone network. Most importantly, we encode the object detection results into the BEV representation to reduce the volume of data transmission and make real-time collaboration between autonomous vehicles possible. The output of this process, the BEV map, can also be used as direct input to most route planning modules. Numerical results show that this novel method can increase the accuracy of object detection by cross-verifying the results from multiple points of view. Thus, in the process, this new method also reduces the object detection challenges that stem from occlusion and partial occlusion. Additionally, different from many existing methods, this new method significantly reduces the data needed for transfer between vehicles, achieving a speed of 21.92 Hz for both the object detection process and the data transmission process, which is sufficiently fast for a real-time system.**

***Index Terms*—Autonomous vehicle, object detection, vehicle-to-vehicle communications.**

I. Introduction

THE goal of an autonomous driving system (ADS) is for the autonomous vehicle (AV) to operate without additional human input [1]. As such, the AVs can leverage a range of sensors for environment perception as well as receiving and integrating other vehicles' perceptions for predicting and planning their navigation.

AVs have the potential to change the future of transportation by leading to safer roads, less traffic congestion, reduced parking, and positive economic impacts on society [2]. However, there are multiple barriers toward autonomous driving, with the most notable one being the safety of passengers. In this paper, our goal is to increase the safety of ADS by leveraging V2V communication to achieve a more complete perception of the environment, especially for occluded objects and blind spots.

In an ADS, the AV detects the environment using its sensors. The two most notable sensors used for object detection are camera sensors and light detection and ranging sensors (LIDARs). Camera data are utilized to mimic the human visual ability to see the world. However, this approach has problems in detecting range and distance, a crucial characteristics of vehicular sensing. Monocular stereo vision (taking pictures with one camera) is unable to detect accurate range and distance. Binocular stereo vision (two cameras) and multi-view stereo vision can use pictures from different angles to form 3D objects and their ranges but matching corresponding points is difficult or computationally expensive. On the other hand, LIDAR can directly obtain the distance information of vehicles, pedestrians, obstacles, and road infrastructure. [3]

In ADS, one of the biggest problems is occlusion [4]. A critical challenge of object detection, particularly for the ADS, is the problem of blind spots and occlusion, which are restricted by the line of sight and field of view of AVs.When objects are partially or entirely occluded from the Line-of-Sight of the camera and LIDAR) sensors, it is difficult and even impossible for most deep learning neural networks to detect and track the occluded objects [5]. Therefore, the AVs cannot detect them in time to plan a new route and avoid collisions, which can lead to hazardous situations for passengers. For example, in Fig. 1, the ego vehicle (EV, vehicle A) cannot "see" the bicyclist using camera or LIDAR as the bicyclist is occluded by the bus. However, vehicle B on the other side of the road can easily see this bicyclist and send the information to the EV. As such, the EV A can avoid a potential accident in case the bicyclist tries to cross the road.

Collaborative Vehicle-to-Vehicle (V2V) communication can be a potential solution for these occlusion scenarios as multiple AVs can communicate for a more comprehensive perception of the environment. A V2V network is a system designed to transmit safety information among AVs to facilitate warning about impending crashes. Thus, an AV can leverage the information from other vehicles in the fleet to be aware of the occluded objects and plan for a new and safer navigation route.

In a V2V network, the AVs' sensors collect the environment information from different perspectives and integrate them together which can minimize or even eliminate the occlusion problem. However, sending a data stream of raw data collected from all sensors (camera, radar, LIDAR, and GPS signal) is inefficient and unscalable due to the huge volume (large datasets from sen-

Manuscript received 14 December 2021; revised 28 July 2022; accepted 7 December 2022. Date of publication 3 April 2023; date of current version 19 September 2023. The work of Hua Fang and Honggang Wang was supported by NSF under Grant ECCS 2010366. The review of this article was coordinated by Prof. Shucheng Yu. *(Corresponding author: Honggang Wang.)*

Hieu Ngo and Hua Fang are with the University of Massachusetts Dartmouth, North Dartmouth, MA 02747 USA (e-mail: honggang.wang@yu.edu; hfang2@umassd.edu).

Honggang Wang is with the Katz school of Science and Health, Yeshiva University, New York 10016 USA (e-mail: honggang.wang@yu.edu).

Digital Object Identifier 10.1109/TVT.2023.3264020

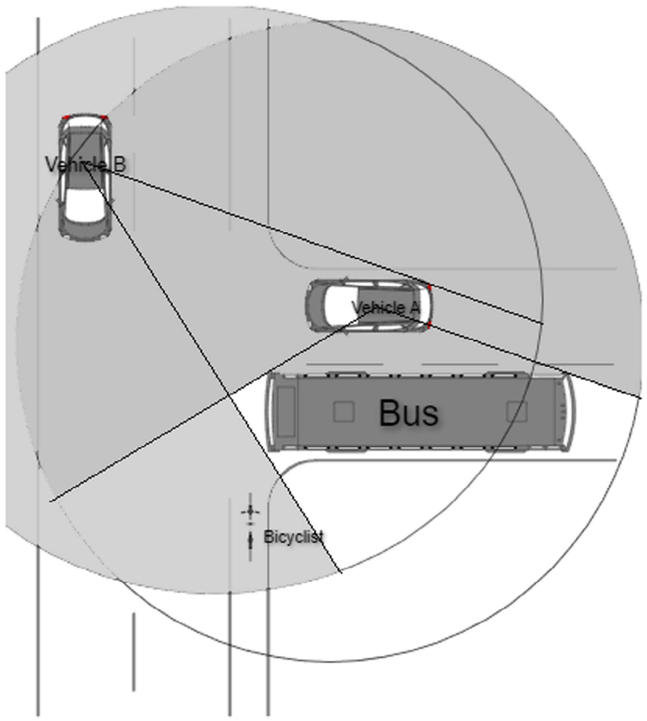

Fig. 1. A bicyclist is occluded by the bus from vehicle A point of view but can be sensed by vehicle B on the other side of the street.

sors, especially LIDAR), variety (incorporating multiple types of sensors), and the required velocity (real-time system) of data in ADS [6]. In the KITTI dataset, the laser scanner pins at 10 frames per second and captures approximately 100 k points per cycle [7]. The cameras (4 cameras) in the KITTI dataset are also triggered at the same rate as the laser scanner, 10 frames per second, with the images' dimensions of 1382 x 512 pixels. In total, if the AVs collect only the LIDAR and cameras data with the same settings, the total data size to transfer the data from one AV to another goes up to 26.7 GB/s

Furthermore, the fusion of multiple point cloud data is not a trivial problem as it requires precise location in the 3D space of LIDAR scanner to perform the transformation of the LIDAR data from different Point Of Views (POV) into one POV [8]. Therefore, to keep the benefit of collaboration in real-time V2V systems, we propose a Real-time Collaborative Vehicular Communication (RTCVC) framework. RTCVC is a BEV map-based LIDAR encoding to communicate the object detection results of vehicles in the same fleet. This 2D representation of LIDAR can provide accurate depth information from LIDAR technology while still accessing the maturity of established, well-trained RGB image-based object detection models in the field. Furthermore, the compactness of this representation allows vehicles to communicate effectively in real time and BEV maps also serve as direct input to most route planning modules without additional complex modifications [9], [10], [11]. All these factors make our proposed method effective in a real-time system. The technical contribution of this paper is proposing a method for combining object detection results collaboratively between AVs to enable real-time collaboration of V2V system.

Our paper is structured as follows: Section II presents the background and related work of object detection in autonomous vehicles; Section III describes our proposed method, RTCVC, including the communication scheme and the object detection modules; Section IV tests the accuracy and speed of our proposed method using simulations and experiments; and Section V discusses and concludes this paper.

II. Background

Object detection is a technology in computer vision that deals with detecting instances of objects such as vehicles, pedestrians, and buildings in digital images or LIDAR images [12]. It has a critical role in scene understanding and environment perception. Therefore, object detection has been widely used in many fields, especially for ADS.

A. *Object Detection*

An object detector typically has two layers: the backbone network and the baseline [13]. The first layer is the backbone network (feature extractor). A backbone network takes images as input and outputs the feature maps of the corresponding input image [14]. It is usually an object classification network minus the last fully connected layers. For backbone networks, there is a tradeoff between accuracy and efficiency. Deeper and densely connected backbones such as Resnet [15], AmoebaNet [16], and ResNeXt [17] provide better accuracy, whereas shallower and sparsely connected backbones such as MobileNet [18], MobileNetV2 [19], ShuffleNet [20], and Xcepttion [3] provide faster inference time. Additionally, improved and newly designed versions of basic classification networks are also utilized to meet specific requirements of object detection tasks, such as DetNet [21] and Fishnet [22].

After the backbone is the feature selection process. Feature selection is the process of selecting a subset of input variables. This dimensional reduction help improve the data redundancy, the computation load, over-fitting, and the accuracy of the machine learning methods [23].

Feature selection in LIDAR images would remove variables that are highly correlated [24], [25]. For example, in the point cloud LIDAR image, a few thousand data points are reflected from the same vehicle. Using features extracted from these points can better represent this vehicle and save memory as well as processing time. The dependent points of this feature provide no extra information about data and only serve as noises for the object detection models. By removing the dependent points, we can significantly reduce the amount of data and improve object detection performance. Additionally, contrary to highly correlated data points, there may exist data points that have no correlation to the environment perception in ADS [26]. Thus, these variables also serve as noises and introduce bias into the system. Therefore, applying feature selection helps remove redundant data and minimize the amount of important data to be transmitted during V2V communication.

There are two main categories of object detectors: two-stage detection with a "coarse-to-fine" process and one-stage detection with a "complete in one step" process [27]. In a two-stage detector, The two stages are Region of Interest (ROI) proposal generation and feature extraction by ROI Pooling (ROIPool). In the second stage, the features are extracted by ROIPool from the candidate boxes to be used in classification and bounding-box regression. The first stage is called a Region Proposal Network (RPN), which proposes the object bounding boxes. Some works that represent the development of CNN-based two-stage detectors are MFR-CNN [28], FII-CenterNet [29], R-CNN [30], Fast R-CNN [31], Faster R-CNN [32], and Mask R-CNN [33].

Contrary to the two-stage detector, a one-stage detector directly proposes the predicted boxes from the input and does not have a region proposal stage. Examples of the one-stage detectors used in smart vehicular system includes [34], [35], SSD [36], DSSD [37], YOLO [38], YOLO V3 [39], and RetinaNet [40]. In conclusion, the two-stage detectors usually have higher localization and object detection accuracy, while the one-stage detectors are more time efficient and have higher inference speeds.

B. Related Work

For point cloud data, there are different strategies for 3D object detection, including multi-view, voxel, and point-based methods.

Multi-view methods work by integrating multiple sensors. MV3D [41] merges the BEV and front view from LIDAR data with the RGB image to generate 3D bounding box predictions. AVOD [42] improves upon MV3D by merging the features in the region proposal network (RPN) phase instead of the refinement phase. Some limitations of these methods include detecting small objects and detecting multiple objects in the same direction.

Some methods used voxel-based representation [42], [43], [44], [45]. In [45], the non-empty voxels are encoded with statistical quantities of the points within the voxel. On the other hand, [46] uses binary encoding for each voxel. PIXOR [44] encodes the voxel with occupancy. Whereas, SECOND [47] uses sparsely embedded convolutional layers for parsing the voxel representation. Aside from this, pseudo-images are also used as the representation after voxelization in PointPillars [48]. These methods all project the sparse point cloud into a more compact voxel representation. The drawback is that the point number in each voxel is limited, thus leading to information loss.

There are also many two-stage detectors with high accuracy. Fast Point R-CNN [49] uses the two-stage detector framework to utilize the volumetric representation for initial predictions and then further refine them with a raw point cloud. STD [50] generates proposal features, uses Pointspool to transform them into a more compact representation and predicts the 3D bounding boxes in the second stage. The high accuracy of these methods comes from a better recall rate. However, due to a longer inference time, they are not viable in a real-time system. Therefore, in this paper, we propose a one-stage framework with better accuracy and time efficiency.

Additionally, data representation and compression are also other important components, which can determine whether an ADS can be collaborative in real time [51]. Therefore, we also propose the communication scheme and data encoding associated with our method, RTCVC, to enable a real-time collaborative autonomous vehicular system.

Algorithm 1: BEV Map Matching.

```
Input  : Ego vehicles' BEV map M; m pairs of BEV
         maps (H, W, 3) and their GPS location
1 Use the transformation matrix T on all the transferred
    BEV map $M_i$
2 while there are non-stitched BEV maps $M_i$ do
3     Let M' be M
4     Adding the additional part of $M_i$ to M' to update
        M'
5     Cross verify all similarly detected object and their
        new confidence interval
6     Mark any detected object from $M_i$ that is different
        from M' for further processing in the route
        planner
7 end
Output: Return M'
```

III. Methodology

As mentioned, the proposed method in this paper is to utilize the advantage of Bird's-Eye-View (BEV) map in collaborative communication to help in occluded object detection, which is an important challenge in ADS.

A. BEV Map-Based V2V Collaboration

A BEV map is a map that represents the top-down view of the surrounding environment of the ego vehicle (EV). In a typical autonomous driving stack, the behavior prediction and planning are generally completed using the BEV map because height is less important whereas BEV can represent most of the information needed for an AV. A BEV map also is much smaller in size compared to the LIDAR point cloud and will be communicated much faster with other AVs in the network.

The BEV map is encoded by height, intensity, and density, where each point on the 2D grid correlates to a $0.1\ \text{m} \times 0.1\ \text{m}$ cell. The point cloud data contains a set of data points in a 3D space, which is collected using LIDAR. These data can be divided into a 2D grid of 0.1×0.1 m cell. As such, The height feature for each cell of the BEV map is calculated as the maximum height of the points in a cell. Similarly, the intensity is the reflectance value of the point with the maximum height. Finally, the density indicates the number of points in each cell. Therefore, a BEV map have dimensions similar to that of an RGB image, with height and width but instead of red, green, and blue value, it has height, intensity, and density value. Thanks to accurate depth information from LIDAR, the point cloud contains the accurate location of all points in the cloud relative to the ego vehicle. The depth information is preserved during the process of encoding the BEV map.

The method we propose in this paper is called RTCVC (Real-time Collaborative Vehicular Communication), which utilizes the compact nature of BEV maps to communicate the

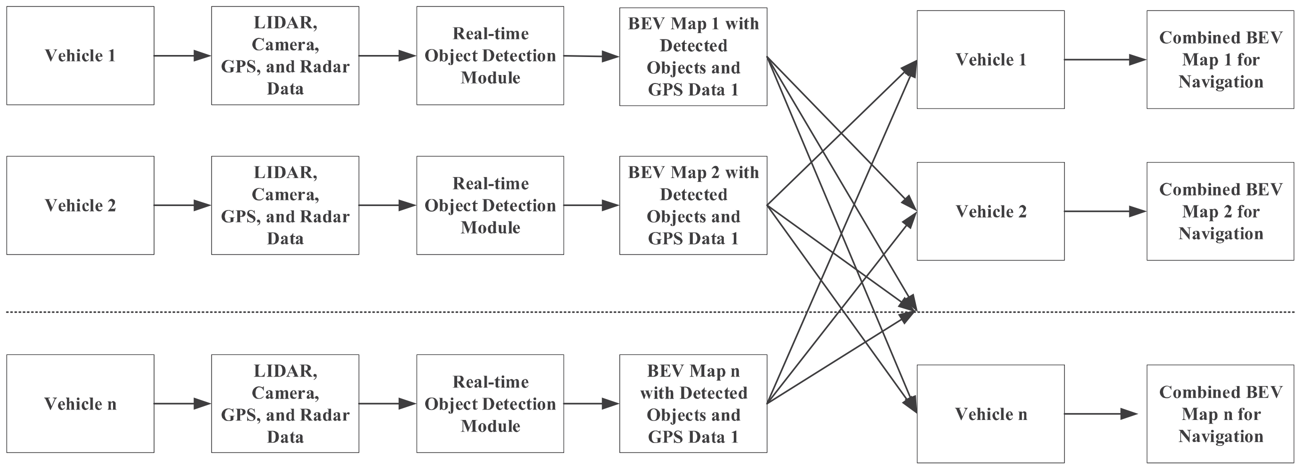
Vehicle 1
LIDAR, Camera, GPS, and Radar Data
Real-time Object Detection Module
BEV Map 1 with Detected Objects and GPS Data 1
Vehicle 2
BEV Map 2 with Detected Objects and GPS Data 1
Vehicle n
BEV Map n with Detected Objects and GPS Data 1
Combined BEV Map 1 for Navigation
Combined BEV Map 2 for Navigation
Combined BEV Map n for Navigation

Fig. 2. Proposed RTCVC BEV framework. This graph illustrates the pipeline of the object detection process of the vehicle in the network. A vehicle senses the environment using its sensors. These data are used as input for the RTCVC object detection module, which outputs a BEV map M encoded with the detected objects and the vehicle's GPS location. Each vehicle will then send this BEV map M to all neighboring vehicles in the network. After receiving these BEV maps, the ego vehicle applies Algorithm 2 to combine the BEV maps with the cross-verification of the detected objects.

Algorithm 2: return BEV Map M' With Detected Objects Use for Navigation Planning.

```
Input: LIDAR data at time t; transferred BEV maps
  from collaborative vehicles with object detections
  results and their GPS location received at time t
1 Use deep learning-based object detection model to
  detect vehicles, pedestrians, cyclists, ... using LIDAR
  data.
2 Make a BEV map M using the LIDAR data and object
  detection results.
3 Using Algorithm 1 to apply BEV map stitching to get
  the final BEV map M'
Output: Return BEV map M' with detected objects used
  for navigation planning
```

environmental perception of neighboring vehicles. The RTCVC framework is a framework for collaborative object detection in V2V networks. When an AV is isolated, there is no collaboration. The collaboration starts when vehicles from the same network are in proximity to each other, which enables communication. The communication method can be done using millimeter wave frequency bands for faster throughput. After the communication starts, the RTCVC collaborative framework can be activated. Each vehicle in the network still runs its own object detection modules. The process of RTCVC is as follows:

- *Step 1:* Each AV i uses its own collected LIDAR point cloud data and projects it onto the 2D grid to create a BEV map.
- *Step 2:* The BEV map is used as input to detect objects and predicts bounding boxes for these objects
- *Step 3:* The bounding boxes are encoded into the BEV map
- *Step 4:* Each AV i sends a copy of this BEV map to all AVs in the network along with their GPS location.
- *Step 5:* Each AV i uses the collection of its own BEV map along with these newly acquired m pairs of BEV maps plus GPS locations of the sending vehicles to make a final BEV map
- *Step 6:* Each vehicle uses its own final BEV to make a navigating decision.

This process is also described in Algorithm 2 and Fig. 2 .

There are two steps in the process of combining the BEV maps, which are described in Algorithm 1. The first step is stitching the additional 2D area of the BEV map to the main map. The second step is cross-verifying and/or adding additional detected objects from other BEV maps.

In the first step described in Algorithm 1, we utilize the GPS signal to align the maps together. Let x and y be the coordinates of the EV in the BEV map. $\{x_i, y_i\}$ is the set of coordinates of the set of vehicles i in the network. Let M be the BEV map of the EV with the vehicle's coordinates (x, y) at the center (0, 0) of the map and $\{M_i\}$ be the set of BEV maps transferred to the EV. To combine these BEV maps, we apply the transformation matrix T:

$$T = \begin{bmatrix} 1 & 0 & -x_i \\ 0 & 1 & -y_i \\ 0 & 0 & 1 \end{bmatrix} \quad (1)$$

To each BEV map M_i. After this, we will stitch the additional area to the BEV map. Specifically, let's define the main BEV map with the dimensions of (H, W, 3) where H, W are the height and width of the map, and the 3 layers are the height, intensity, and density of an area on the map. Additional BEV maps will provide a bigger area of perception, giving the main BEV map a bigger H', W'.

In the second step, we cross-verify and/or add the additional detected objects to the main BEV map. After the object detection step, the BEV map contains not only the top-down view of the environment but also the detected objects' center coordinates; length, width, and height; their heading angle, and semantic label if provided. The cross-verification process serves to increase the confidence interval. Especially, the difference in detected results such as that of occlusion or misidentification will be used in the route planning process. The route planner can choose to either slow down or do other precautious actions in these cases to minimize the risk of accidents while waiting for new or updated information on the environment.

Thus, this method works very well for the partially and entirely occluded objects from the ego vehicle when it has extra information from surrounding vehicles. Particularly, in situations where occlusion happens due to Non-line-of-sight (NLOS) when a vehicle cannot see the whole or part of an object, but another vehicle can from a different perspective, the BEV map from the transmitting vehicles can be integrated into the ego vehicles' BEV map and help reinforce or add the detection of the occluded object to the main BEV map. Not only does it help in occlusion, but this method can also increase the detection accuracy of vehicles in cases where the accuracy is not high.

Moreover, by using the BEV map (2D data) and not point cloud data (3D data), we also make it easier to fuse the views from multiple data sources, which can be extremely difficult and computationally expensive in the point cloud data case [52].

In the numerical analysis section, we will further explore the effectiveness and speed of the proposed RTCVC method using real data and experiments. Overall, using empirical methods, the RTCVC method showed that it increases the accuracy of object detection, especially for occluded objects. Using experiments, we also show that the RTCVC method significantly reduce the transmission load with less than 400 times the transferred data of compressed LIDAR data (LAZ compression). Thus, making this method possible in real time, with a frequency of 21.92 Hz.

B. RTCVC BEV Map-Based Object Detection

For object detection, the proposed RTCVC can work with a multitude of object detection modules that are suitable to the ADS. For this proposed method, we use a RTCVC BEV map based object detection built upon a Resnet-based Keypoint Feature Pyramid Network (KFPN) in a RTM3D model for object detection [53], [54]. The difference of our proposed RTCVC method from the RTM3D model is that instead of using the RGB images from the camera as the input, we input the BEV map that is encoded by height, intensity, and density of the 3D LIDAR point clouds. This help leverage the precise depth information of the LIDAR data using a 2D image object detection module. Thus, we can have a faster inference time than using the point cloud data while having a higher accuracy of well-trained 2D object detection modules.

The backbone network we use is Resnet-18 [15] with a down sample factor S = 4. The input is a single BEV map $M \in \mathbb{R}^{W\times H\times 3}$. The feature maps of the low levels are also concatenated and a 1×1 convolutional layer is added for dimension reduction before the upsampling layers. The bottleneck of Resnet is then upsampled by three bilinear interpolations and a 1×1 convolutional layer. After these upsampling layers, the resulting channels are 256, 128, and 64.

The Keypoint Feature Pyramid Network (KFPN) [53] is used to detect scale-invariant key-points in the point-wise space. A normal Feature Pyramid (FPN) extracts the features from a pyramid of features map at different scales and use them for object detection. A keypoint FPN are used to detect multi-scale 2D box in different pyramid layers, we need to use KFPN to detect mutli-scale 3D box at different scales. Assuming that we have F scale feature maps. For each scale feature map f, we resize f to a scale feature map with a maximal size $\hat{f}$. Thus, all F scale feature maps f have the same size $\hat{f}$. We can obtain the soft weight of the key point feature map by applying the softmax operation. Finally, the scale score is the linear sum of all the resized scale maps' soft weights:

$$S_{score} = \sum_{f} \hat{f} \odot softmax(f) \tag{2}$$

The output of our model includes the center coordinates of the detected objects; the length, width, and height of the bounding box; and the heading angle of the bounding box. The goal of our BEV-based KFPN is to detect these 7 features of an object in the scene to detect the bounding box surrounding the object.

The detection head uses the center coordinates (c_x, c_y, c_z), the length, width, and height (l, w, h) of the bounding box, and the heading angle in radians (θ) of the bounding box as the target. The main center key point is used to connect all features. The heatmap of the main center is defined as $C \in [0, 1]^{W/S\times H/S\times G}$ where G is the number of object categories. This heatmap of the main center is trained using focal loss. The focal loss function applies a modulating term to the cross entropy loss to focus on learning the hard negative example. Thus, it reduces the affect of class imbalance during training, such as that in object detection [40]. The heading angle θ defines the rotation of an object. To find the heading angle, we use the Multi-Bin based method [55] for orientation regression. One bin is used to classify the probability of cosine and sine of local angle. This bin is used to generate the feature map $O \in \mathbb{R}^{H/S\times W/S\times 8}$ with two bins. The dimension $D \in \mathbb{R}^{H/S\times W/S\times 3}$ of the objects are trained using the balanced $L_1 Loss$ [56]. Finally, the z coordinates $Z \in \mathbb{R}^{H/S\times W/S\times 1}$ are also regressed using the balanced $L_1 Loss$.

The training of the method then works to minimize the focal loss and detect the keypoints. As a result, for each object in a BEV map M, there are 7 features representing this object, including its categories, center coordinates, length, width, and height of the bounding box, and the heading angle of the bounding box.

The novelty of this method is utilizing the efficiency and maturity of image-based object detection methods for LIDAR images. The precise depth information of the LIDAR helps increase the accuracy of the detection. Most importantly, the compactness of BEV maps reduces the bulk of object detection computation as well as communication cost. Thus, we can have a fast enough inference time and accuracy to use in a real-time ADS.

IV. Numerical Analysis

In this section, we test both the effectiveness of our proposed method in detecting occluded objects as well as its inference speed using simulations and experiments.

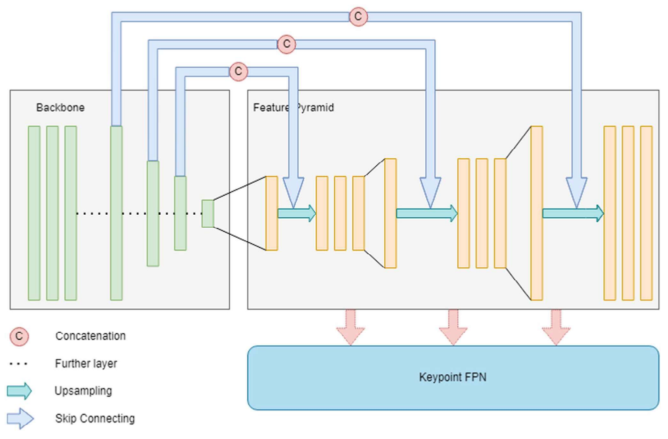

Fig. 3. Object detection model that takes BEV map as input. The backbone networks process the input BEV maps and extract features. The feature Keypoint Feature Pyramid selects features and passes them for detection.

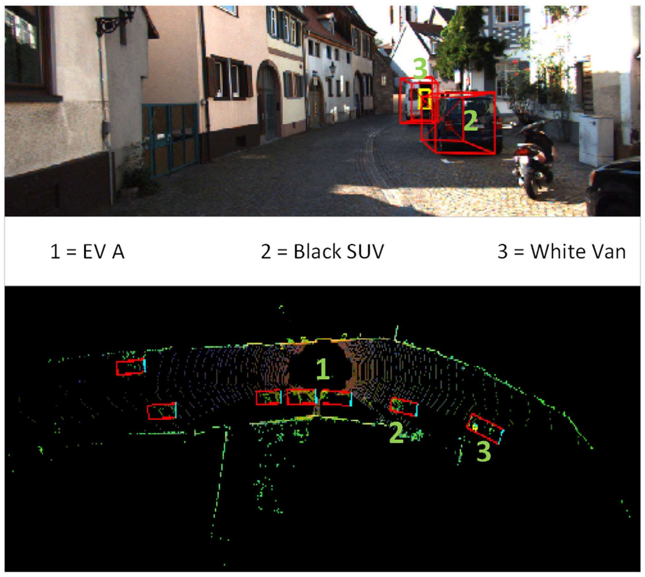

Fig. 4. Example case where the bicyclist is occluded from one vehicle's view but not the others'.The EV A is traveling in a cured road. The EV can detect the Black SUV and the white van in the front. This frame is taken approximately 0.7 s earlier than Fig. 5, which makes the EV to be behind vehicle B in a real world scenario.

A. *Occluded Object Detection*

In this section, we aim to test the use case of BEV map-based collaborative detection in V2V communication, with a focus on solving partial occlusion and full occlusion due to non Line-of-sight(LOS). Particularly, we focus on the scenarios where the occlusion appears because of other objects as opposed to occlusion due to weather conditions such as snow, rain, and fog. The data we are using come from the KITTI dataset [7]. To simulate the collaborative communication between two AVs, we take the LIDAR point cloud and image instance of the same vehicle at two different time stamps. In this way, we can imagine these as two vehicles operating on the same road at the same time. The time stamps distance is chosen such that, if these are two separate vehicles, they would be in line-of-sight and within the range of 10- 30 m. Thus, we can ensure that the cases discussed here are realistic and the communication be- tween the vehicles can be assumed due to the line- of-sight and distance.

In this simulation, we use our proposed method RTCVC described in section 3 to detect the object using a compact representation of the LIDAR point cloud — BEV map (H, W, 3). For demonstration, the detected objects are projected onto the image for better understandability and interpretability. The BEV map is also included and used as the mean to communicate the detection results between vehicles. In Figs. 4 and 5, we examine the first case where the frame in Fig. 4, including the image and BEV map, is taken approximately 0.7 s earlier than Fig. 5. Therefore, in a real world scenario, the vehicle A would be right behind of vehicle B. Assume vehicle A to be the ego vehicle (EV), we observe that it cannot detect the pedestrian who is standing right behind the white van. This situation can be very dangerous because the time window for decision is very small if the pedestrian decides to walk toward the driving lane. However, from the front vehicle B point of view, it can clearly detect the pedestrian and noted it on the BEV map as "Pedestrian". With the information given by the BEV map of the front vehicle, the ego vehicle, EV A in Figs. 3 and 5, has more time and space to make a better decision on its speed and direction in the next 0.5-1 s. In Figs. 6 and 7, similarly, we examine the case where the first frame in Fig. 6 is taken 1.1 s earlier from the frame in Fig. 7. In this situation, vehicle B detect a bicyclist riding in the opposite direction on the same road whereas the EV A cannot. Similarly in this scenario, collaboration between the two AVs is much needed to avoid accident. Furthermore, in a real world situation where the front vehicle is physically in front of the ego vehicle, it is even more difficult to detect the bicyclist in time before he gets near.

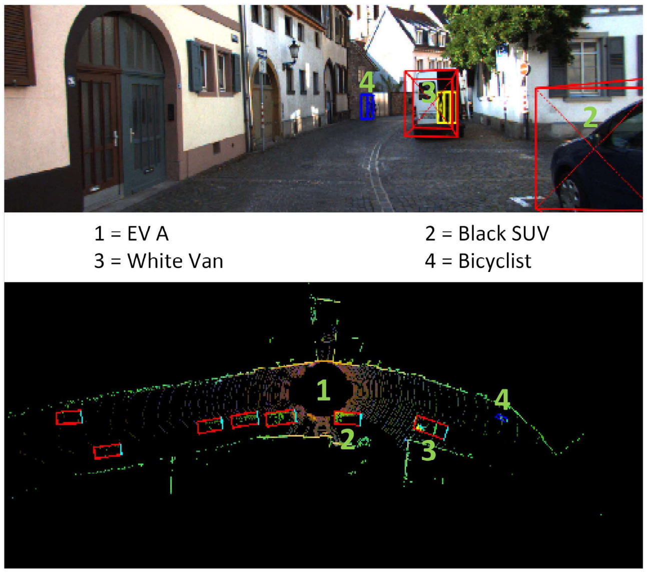

Fig. 5. Vehicle B is traveling in the front of EV A. Both vehicle A and B can detect the black SUV, the white van on the right. However, EV A cannot detect the bicyclist coming from the front, occluded from the curved road. Vehicle B can detect the bicyclist and transfer this information through BEV to vehicle A.

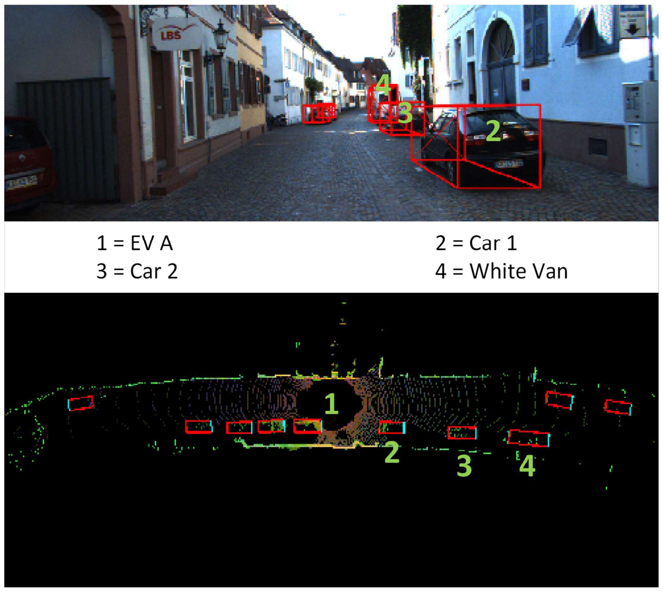

Fig. 6. Example case where the pedestrian is occluded from one vehicle's view but not the others'. The EV A is traveling in a straight road. The EV can detect the car 1, car 2, and the white van ahead. This frame is taken approximately 1.1 s earlier than Fig. 7, which makes the EV to be behind vehicle B in a real world scenario.

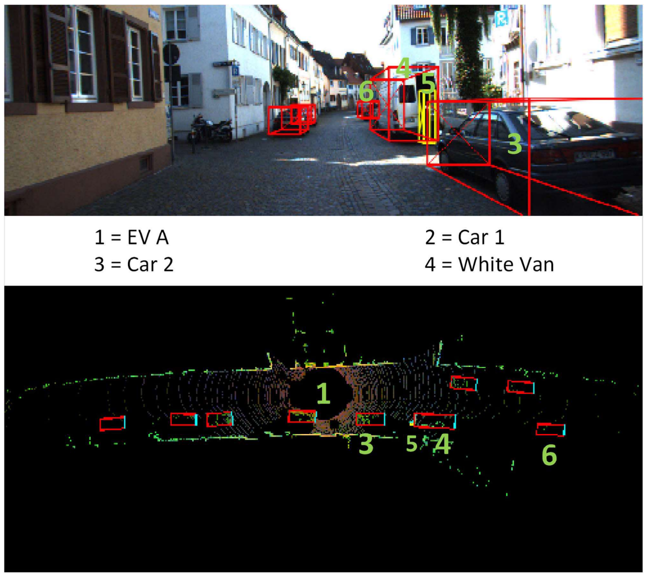

Fig. 7. Vehicle B is traveling in front of the EV A. Vehicle A can detect car 1, car 2, and the white van on the right. Vehicle B, however, also detect a pedestrian behind the white van and car 3 further in the front.

Based on these scenarios, we can see that the collaboration using detection results encoded in BEV map can increase the accuracy of detection results and help tremendously in occlusion cases.

TABLE I
ACRONYMS

ADS	Autonomous Driving System
AV	Autonomous Vehicle
V2V	Vehicle-to-Vehicle
BEV	Bird's-Eye-View
EV	Ego Vehicle
FPN	Feature Pyramid Network
KFPN	Keypoint Feature Pyramid Network
RGB	Red, Green, Blue
ROI	Region of Interest
ROIPool	ROI Pooling
RPN	Region Proposal Network
LOS	Line-of-sight
NLOS	Non-Line-of-sight
LIDAR	Light Detection and Ranging
POV	Point Of Views
T	Transformation Matrix used for BEV maps
H	Height
W	Width
S	Down sample factor
f	scale feature map
$\hat{f}$	scale feature map with a maximal size
F	number of scale feature maps
M	BEV map before stitching
M_i	transferred BEV from collaborating vehicle
M'	Final BEV map after stitching
c_x, c_y, c_z	center coordinates of bounding box
l, w, h	length, width, and height of the bounding box
θ	the heading angle of detected object
C	Heatmap of the main center
G	Number of object categories
O	Feature map for heading angle regression
D	dimensions of objects
Z	z coordinates

B. *Processing Time and Data Size for Transmission in RTCVC*

In this section, we simulate the processing time and data size for transmission in the scenarios in Section IV and show that our method can work in a real-time system.

Besides detection accuracy, another critical factor for ADS is the processing and transmission time. The speed must be fast in order to satisfy the requirements of a real-time driving system. Thus, we aim to provide numerical analysis on the processing time of BEV map-based object detection as well as the data size and transmission time of BEVs map.

In Table I, we compare the file size of the raw camera data, the LIDAR data, the BEV map, and our proposed method of sending the BEV maps with camera data used in the simulation above. These numbers are based upon the KITTI dataset that we used to run our tests. The BEV map keeps the important depth information from the LIDAR data. On the other hand, the detection results from the camera data can provide additional information to the ego vehicle such as traffic sign, road surface markings, etc..

For simulation, we consider both use cases of 4G and 5G network. Ideally, the V2V network will use a fast 5G network to communicate to each other. However, in cases of area where 5G

TABLE II
FILE SIZES AND TRANSMISSION TIME COMPARISON IN 4G AND 5G NETWORK FOR BEV MAPS, CAMERA DATA, AND LIDAR DATA

Method	File size (Mbs)	Transmission using 4G network (14 Mbps)	Transmission using 5G network (100 Mbps)
BEV maps	1.1	0.0758 s	0.0106 s
Camera (4 cameras)	8.5	0.6065 s	0.0849 s
LIDAR point cloud	28000.0	2000.0000 s	280.0000 s
LAZ LIDAR	4000.0	285.7000 s	40.0000 s
Ours (BEV map + Camera)	9.6	0.6823 s	0.0955 s

network is not available, we will also make a simulation using the 4G network. In this simulation, we use the average speed for a 4G network at 14 Mbps. On the other hand, a 5G network speed can range from 50 Mbps to over 1 Gbps. For this paper, we use a sub-6 GHz 5G (mid-band), the most common band, at 100 Mbps for this numerical experiment. The BEV map size is the same as that of a RGB image, with the size of 1382×512 pixels. Similarly, the data size of the cameras' images is also calculated by multiplying the size of a 1382×512 pixels image with 4 to match with the setup of the KITTI dataset. Finally, the LIDAR scanner is capturing approximately 100 k points per cycle. Using the LAS formats, it's approximately 28 bytes per point. Additionally, we can also compress it using the LAZ format (laszip), which has a compression ratio of 7:1. With these information, we can calculate the transmission time needed for each data type, listed in Table I. Using this setup, we can easily see that sending the raw LIDAR is impractical as even in a faster 5G network with a speed of 100 Mbps and the LIDAR data is compressed, it still takes 40 seconds to send the full data. In the KITTI setup, both the camera and laser scanner capture information at a rate of 10 Hz, or 0.1 s between each frame. Therefore, it is impossible to have proper collaboration between using the compression methods above when the transmission time significantly outweighs the capture speed.

In Table II , we compare the processing time of LIDAR-based object detection methods including IPOD [57], F-ConvNet [58], STD [50], Point RCNN [59], Fast Point R-CNN [49], SECOND [47], HRI-Voxel FPN [60], Point Pillars [48], PIXOR++ [61], and HVNet [62] with our method, an object detector using the RTM3D network structure with BEV map input, BEV map input, Resnet18 backbone, and BEV-map based collaborative perception. On top of this, we also simulate the process of transmitting the detection results in a BEV map, which includes both the encoded detection results and the GPS location. We can see that even with the transmission of the BEV map in a 5G network, our method still has a competitive processing time, at 0.0456 seconds, faster than all the compared two-stage detectors as listed in Table II. The methods we compared with include IPOD [57], F-ConvNet [58], STD [50], PointRCNN [59], Fast Point R-CNN [49],.Furthermore, it is also very close to the one-stage detector such as SECOND [47], HRIVoxelFPN [60], PointPillars [48], PIXOR++ [61], HVNet [62], and RTM3D

TABLE III
PROCESSING TIME COMPARISON. ALL METHODS USE LIDAR INPUT. THE METHODS ARE DIVIDED INTO TWO TYPES: TWO-STAGE AND ONE-STAGE DETECTORS

Type	Method	Time
Two-stage	IPOD [57]	0.2000 s
	F-ConvNet [58]	0.4760 s
	STD [50]	0.0800 s
	PointRCNN [59]	0.1000 s
	Fast Point R-CNN [49]	0.0600 s
	SECOND [47]	0.0500 s
One-stage	HRI-VoxelFPN [60]	0.0200 s
	PointPillars [48]	0.0235 s
	PIXOR++ [61]	0.0286 s
	HVNet [62]	0.0322 s
	RTM3D (Resnet18) [53]	0.0350 s
Proposed method	RTVC	0.0456 s
	RTCVC +Camera	0.1305 s

[11] methods with the fastest one at 0.02 s and slowest one at 0.05 s.

In the KITTI dataset, the laser scanner spin at 10 frames per second. The cameras are also triggered at 10 frames per second to match the laser scanner. Therefore, each frame of image is 0.1 s away from each other (10 Hz) and any method that needs more than this time will not be viable for a real-time system. Using the results from Table II, it takes only 0.0456 s for our method to detect the object and send the BEV map results to another vehicle. Therefore, it is completely practical to use our method in a real-time system with a setup like that of the KITTI dataset. This method can also potentially work at the frequency of 21.92 Hz, which means the laser scanner and the cameras can capture the environment at a higher speed, up to double the current capture rate of KITTI.

Overall, compared to other methods, our proposed method RTCVC can achieve a fast reference time for a real-time system, high accuracy, and most importantly, much better accuracy at detecting occluded objects, which is a challenge for most object detection modules.

V. CONCLUSION

In this paper, we proposed RTCVC, a collaborative environment perception method for autonomous driving system. RTCVC uses BEV map-based object detections and encode the detected results onto the BEV to communicate these results with other vehicles. The BEV map is a compact 2D representation of LIDAR point cloud. This representation helps us to utilize the accurate depth information from LIDAR, which is difficult for camera data. The 2D form also enable us to use mature, well-trained image-based object detection network such as Resnet. Furthermore, since BEV map is a popular input for many route planning networks, our method does not need to modify the outputs of the network before passing them to route planner. Due to a significant reduction in volume of communication, RTCVC can establish real-time collaboration between autonomous vehicles, which is difficult in other methods due to the transmitting delay between vehicles. Thus, our method helps reduce the problem of occlusion and partial occlusion in the autonomous driving system, a challenging problem in ADS without collaboration. Using simulation, we show that this method can increase the accuracy of object detection, especially in occlusion cases due to NLOS. Furthermore, using experiments and the setup in our simulation, we showed that this method can be used in a real-time system, at a reference time of up to 21.92 Hz.

References

[1] C. Berger and B. Rumpe, "Engineering autonomous driving software." 2014. Accessed: Jun. 6, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.1409.6579

[2] D. J. Fagnant and K. Kockelman, "Preparing a nation for autonomous vehicles: Opportunities, barriers and policy recommendations," *Transp. Res. Part A: Policy Pract.*, vol. 77, pp. 167–181, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0965856415000804

[3] Q. Chen, Y. Xie, S. Guo, J. Bai, and Q. Shu, "Sensing system of environmental perception technologies for driverless vehicle: A review of state of the art and challenges," *Sensors Actuators A: Phys.*, vol. 319, 2021, Art. no. 112566. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0924424721000273

[4] K. Wang et al., "The adaptability and challenges of autonomous vehicles to pedestrians in urban China," *Accident Anal. Prevention*, vol. 145, 2020, Art. no. 105692.

[5] A. Wang, Y. Sun, A. Kortylewski, and A. L. Yuille, "Robust object detection under occlusion with context-aware compositionalnets," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 12645–12654.

[6] H. Fang, Z. Zhang, C. J. Wang, M. Daneshmand, C. Wang, and H. Wang, "A survey of big data research," *IEEE Netw.*, vol. 29, no. 5, pp. 6–9, Sep./Oct. 2015.

[7] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[8] S. Kunwar et al., "Large-scale semantic 3-D reconstruction: Outcome of the 2019 IEEE GRSS data fusion contest—Part A," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 922–935, 2021.

[9] S. Aradi, "Survey of deep reinforcement learning for motion planning of autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 2, pp. 740–759, Feb. 2022.

[10] E. Wang, H. Cui, S. Yalamanchi, M. Moorthy, and N. Djuric, "Improving movement predictions of traffic actors in Bird's–Eye view models using GANs and differentiable trajectory rasterization," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 2340–2348, doi: 10.1145/3394486.3403283.

[11] K.-H. Lee et al., "PillarFlow: End-to-end Birds-Eye-view flow estimation for autonomous driving," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 2007–2013.

[12] R. Padilla, S. L. Netto, and E. A. da Silva, "A survey on performance metrics for object-detection algorithms," in *Proc. Int. Conf. Syst., Signals Image Process.*, 2020, pp. 237–242.

[13] L. Liu et al., "Deep learning for generic object detection: A survey," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 261–318, 2020.

[14] R. Zebari, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed, "A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction," *J. Appl. Sci. Technol. Trends*, vol. 1, no. 2, pp. 56–70, 2020.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[16] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 4780–4789.

[17] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1492–1500.

[18] A. G. Howard et al., "Mobilenets: Efficient convolutional neural networks for mobile vision applications." 2017. Accessed: Jun. 6, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.1704.04861

[19] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4510–4520.

[20] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6848–6856.

[21] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun, "DetNet: A backbone network for object detection." 2018. Accessed: Jun. 6, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.1804.06215

[22] S. Sun, J. Pang, J. Shi, S. Yi, and W. Ouyang, "FishNet: A versatile backbone for image, region, and pixel level prediction," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 762–772.

[23] Q. Al-Tashi, S. J. Abdulkadir, H. M. Rais, S. Mirjalili, and H. Alhussian, "Approaches to multi-objective feature selection: A systematic literature review," *IEEE Access*, vol. 8, pp. 125076–125096, 2020.

[24] R. Cheng, R. Razani, E. Taghavi, E. Li, and B. Liu, "2-S3Net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 12547–12556.

[25] M. Oka et al., "Spatial feature-based prioritization for transmission of point cloud data in 3D-image sensor networks," *IEEE Sensors J.*, vol. 21, no. 20, pp. 23145–23161, Oct. 2021.

[26] W. Han, C. Wen, C. Wang, X. Li, and Q. Li, "Point2Node: Correlation learning of dynamic-node for point cloud feature modeling," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 07, 2020, pp. 10925–10932.

[27] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey." 2019. Accessed: Jun. 6, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.1905.05055

[28] H. Zhang, K. Wang, Y. Tian, C. Gou, and F.-Y. Wang, "MFR-CNN: Incorporating multi-scale features and global information for traffic object detection," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8019–8030, Sep. 2018.

[29] S. Fan et al., "FII-CenterNet: An anchor-free detector with foreground attention for traffic object detection," *IEEE Trans. Veh. Technol.*, vol. 70, no. 1, pp. 121–132, Jan. 2021.

[30] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 580–587.

[31] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1440–1448.

[32] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. 28th Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[33] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2961–2969.

[34] C.-T. Lin, S.-P. Chen, P. S. Santoso, H.-J. Lin, and S.-H. Lai, "Real-time single-stage vehicle detector optimized by multi-stage image-based online hard example mining," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 1505–1518, Feb. 2020.

[35] M. Mobahi and S. H. Sadati, "An improved deep learning solution for object detection in self-driving cars," in *Proc. 28th Iranian Conf. Elect. Eng.*, 2020, pp. 1–5.

[36] W. Liu et al., "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2016, pp. 21–37.

[37] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "DSSD : Deconvolutional single shot detector." 2017. Accessed: Jun. 6, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.1701.06659

[38] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 779–788.

[39] J. Redmon and A. Farhadi, "Yolov3: An Incremental Improvement." 2018. Accessed: Jun. 6, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.1804.02767

[40] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2980–2988.

[41] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3D object detection network for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1907–1915.

[42] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. Waslander, "Joint 3D proposal generation and object detection from view aggregation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 1–8.

[43] 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 *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 1355–1361.

[44] B. Yang, W. Luo, and R. Urtasun, “Pixor: Real-time 3D object detection from point clouds,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7652–7660.

[45] D. Z. Wang and I. Posner, “Voting for voting in online point cloud object detection,” in *Proc. Robot.: Sci. Syst.*, vol. 1, 2015, pp. 10–15.

[46] B. Li, “3D fully convolutional network for vehicle detection in point cloud,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 1513–1518.

[47] Y. Yan, Y. Mao, and B. Li, “Second: Sparsely embedded convolutional detection,” *Sensors*, vol. 18, no. 10, 2018, Art. no. 3337.

[48] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, “PointPillars: Fast encoders for object detection from point clouds,” in *Proc, IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12697–12705.

[49] Y. Chen, S. Liu, X. Shen, and J. Jia, “Fast point R-CNN,” in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 9775–9784.

[50] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia, “STD: Sparse-to-dense 3D object detector for point cloud,” in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1951–1960.

[51] H. Wang, H. Ngo, and H. Fang, “Beamforming and scalable image processing in vehicle-to-vehicle networks,” *J. Signal Process. Syst.*, vol. 94, pp. 445–454, 2022.

[52] Y. Lian et al., “Large-scale semantic 3-D reconstruction: Outcome of the 2019 IEEE GRSS data fusion contest—Part B,” *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 1158–1170, 2021.

[53] P. Li, H. Zhao, P. Liu, and F. Cao, “RTM3D: Real-time monocular 3D detection from object keypoints for autonomous driving,” in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 644–660.

[54] N. M. Dung, “Super-Fast-Accurate-3D-Object-Detection-PyTorch.” 2020. Accessed: Jun. 6, 2023. [Online]. Available: https://github.com/maudzung/Super-Fast-Accurate-737 3D-Object-Detection

[55] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, “3D bounding box estimation using deep learning and geometry,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 7074–7082.

[56] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, “Libra R-CNN: Towards balanced learning for object detection,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 821–830.

[57] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia, “IPOD: Intensive point-based object detector for point cloud.” 2018. Accessed: Jun. 6, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.1812.05276

[58] Z. Wang and K. Jia, “Frustum ConvNet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 1742–1749.

[59] S. Shi, X. Wang, and H. Li, “PointRCNN: 3D object proposal generation and detection from point cloud,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 770–779.

[60] H. Kuang, B. Wang, J. An, M. Zhang, and Z. Zhang, “Voxel-FPN: Multi-scale voxel feature aggregation in 3D object detection from point clouds,” *Sensors*, vol. 20, no. 3, 2020, Art. no. 704.

[61] B. Yang, M. Liang, and R. Urtasun, “HDNET: Exploiting HD maps for 3D object detection,” in *Proc. Conf. Robot Learn.*, 2020, pp. 146–155.

[62] M. Ye, S. Xu, and T. Cao, “HVNet: Hybrid voxel network for LiDAR based 3D object detection,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 1631–1640.