Low-Cost LiDAR-Based Vehicle Detection for Self-driving Container Trucks at Seaport

Changjie Zhang[2], Zhenchao Ouyang[1,3(✉)], Lu Ren[1,2], and Yu Liu[1,2]

[1] Hangzhou Innovation Institute, Hangzhou 31000, China
[2] State Key Laboratory of Software Development Environment, BeiHang University, Beijing 100191, China
ouyangkid@buaa.edu.cn
[3] Zhongfa Aviation University, Hangzhou 31000, Zhejiang, China

Abstract. The self-driving technology has been developed rapidly in the past decades, due to new sensors, and car manufacturers have become more open. However, fully self-driving vehicles for the public still has a long way to go. Most studies try to focus on self-driving in special scenes, such as park sightseeing car, express logistics vehicle, sweeper, indoor service robot, and special vehicles in the mining area or seaport area. One of the critical issues is that the cost of a self-driving vehicle should strictly be controlled for commercial uses. This paper presents a low-cost LiDAR-based moving obstacle detection and tracking for self-driving container trucks in the low-speed seaport area. We build a CNN model for obstacle detection with the bird's eye view (BEV) map generated from two low density LiDARs equipped at the head of a container truck. A boosting tracker is used to achieve real-time processing speed on the embedded module of Tx2. Simulation on the collected data shows that our Strided-Yolo model can achieve the highest mAP on the BEV projection map than other models.

Keywords: Obstacle detection and tracking · Autonomous truck · Multiple LiDAR · Deep learning · Seaport area

1 Introduction

In the past two decades, self-driving technologies have attracted interests from both academic researchers and commercial capital, especially from some IT giants, such as Waymo [32] from Google and Uber [1], who have published their self-driving passenger vehicles. However, besides many technical problems, the legal issues related also constrain the accessibility of self-driving vehicles to the public. Recent researches [6,16,19,23,31,35] begins to explore the applications of self-driving in specific scenarios, especially in the closed or semi-closed area with low speed, such as self-driving mine vehicles, container trucks and agricultural vehicles.

H. Gao and X. Wang (Eds.): CollaborateCom 2021, LNICST 407, pp. 451–466, 2021.
https://doi.org/10.1007/978-3-030-92638-0_27

Different from the research phase, a demo self-driving vehicle can be equipped with the best sensors for different targets [2,8]. Considering the commercial usage, the cost of the sensor solution is strictly constrained for large-scale deployment of self-driving vehicle. Selection of related equipment and sensors is a big challenge for container trucks. Moreover, different from normal vision-based detection tasks, the decision system in self-driving relies highly on accurate sensing data of the surrounding environment. Because the traditional camera-based front-view cannot offer accurate location information of on-road obstacles, almost all self-driving systems rely on the Light Detection And Ranging (LiDAR) data. Table 1 compares several of the reported solutions, it can be seen that research projects, such as KITTI, nuScenes and Argoverse use top mounting solutions for small vehicle. For commercial solutions, Waymo uses four short-range LiDAR on each side of the vehicle and a high resolution LiDAR on the top. Audi auto uses 5 Velodyne-16 on the top, and Tusimple highway solution uses two high resolution LiDAR on the each side of the truck head, and one static front-view LiDAR on the top. The mainstream solution is to install multiple LiDARs to ensure safety, but at the expense of sensor costs. Especially for large trucks, the blind area of a single LiDAR top installation scheme is too large.

Table 1. Comparison of the current reported LiDAR solutions.

Dataset/solution	Vehicle	Lidar	Position
KITTI [8]	VW Passat station wagon	Velodyne 64E	Top
nuScenes [2]	Renault Zoe electric car	Velodyne 32	Top
Waymo [15]	Waymo	1 Mid-range & 4 short-range	Top+side
Argoverse [3]	Ford Fusion Hybrids	Velodyne 32 * 2	Top
A2D2 [9]	Audi Car	Velodyne16*5	Top
Tusimple[a]	Truck	4 range LiDAR & 1 static LiDAR static	Top+side

[a] https://www.tusimple.com/

Although the price of current LiDAR sensors dropped quickly in the past several years, the cost for a redundant perception system still very high, especially when some dense LiDAR (such as Velodyne HDL-64E or HDL-32E) is used. Therefore, our system tries to reduce the cost by combining two low density LiDAR (Velodyne VLP-16) as the basic sensor module in our fully self-driving container truck.

This paper presents a deep learning-based CNN model for low-cost LiDAR 3D point cloud vehicle detection. With a series of point cloud pre-processing, such as distance filtering, frame accumulation, ground elimination and bird's eye view (BEV) projection, we can generate stable obstacle features for later detection and tracking. We build a new dataset with dual-LiDAR and GPS-RTK collected with our own self-driving container trucks in a seaport area in China. A well-designed lightweight CNN model is then trained based on this dataset. Both

off-line and on-road tests show that our tracking system can achieve reliable results in real-time.

The contributions of this paper can be summarized as follows:

- A low-cost LiDAR-based obstacle detection and tracking system that uses only two low density LiDAR and GPS-RTK is designed. The system combines traditional point cloud process modules (ground removing and point cloud BEV projection) and CNN model together to achieve high accuracy. The total cost is also reduced.
- A lightweight CNN model is proposed to fulfill the real-time detection task based on the refined BEV map of LiDAR point cloud.
- A new dual-LiDAR based point cloud dataset in the seaport area is also presented based on a novel dataset collected from real seaport in Ningbo.

The rest of this paper is organized as follows. In Sect. 2, we review the LiDAR-based detection and related CNN architectures. We present our sensor solution and the low-cost LiDAR-based vehicle detection and tracking system in Sect. 3. In Sect. 4, we evaluate our system with a real LiDAR dataset collected from a seaport area in the east coast of China. Finally, we summarize the advantages and disadvantages of our system in Sect. 5.

2 Related Work

LiDAR-based obstacle detection in the self-driving area can be divided into two main categories: LiDAR 3D point cloud detection and LiDAR projection based detection.

2.1 LiDAR 3D Point Cloud Detection

The LiDAR 3D point cloud detection model often uses raw LiDAR point cloud or voxelized point cloud as the input to later deep learning models.

The 3D FCN [12] first changes the traditional 2D convolutional network module into 3D full convolutional network module to process the raw 3D point cloud from LiDAR. Therefore, the 3D FCN can use 3D bounding boxes as labels, and predict the target in a 3D manner. Simulation on Kitti dataset [7] shows that the 3D FCN can achieve better performance than traditional point cloud based detection approaches. In this way, the whole training process can be completed in an end-to-end way.

Due to the randomness of LiDAR point cloud, the representation of the same object is much disordered than pixel based pictures. Voxelization is commonly used to reduce this kind of randomness. The VoxelNet [40] first transfers raw point cloud into equally spaced 3D voxels, but the down-sampling process also reduces the features of the point cloud. A voxel feature encoding (VFE) [27] layer that combines with the region proposal network architecture is then presented for 3D object detection. This model can achieve effective representation of vehicles and human beings with a Velodyne HDL-64e.

Due to the huge computation of high dimensionality of point cloud and voxel, PIXOR [37] can achieve real-time 3D object detection through a proposal-free, single-stage detector that outputs oriented 3D object estimates decoded from pixel-wise neural network predictions. This model considers a series of optimization including feature representation, network architecture, time consumption and detection accuracy.

In [13], the authors present a 3D Backbone Network to solve the 3D feature extraction task. With this backbone, it is much easier to extract rich 3D feature maps by using sparse 3D CNN [34] operations, and benefit for later operation such as detection or classification. This module can transfer raw 3D point cloud into multiple 3D images efficiently way. Experiments on KITTI also show their model can achieve a reliable result in detection tasks.

In PointGrid [11], a full CNN of classification network and a U-Net segmentation network are presented for different tasks with raw point cloud input. Both of the two networks begin with a pointgrid [11] input layer that can encode the raw point cloud into multi-dimension matrix for later convolution. This model can achieve state-of-the-art performance in related tasks.

2.2 LiDAR Projection-Based Detection

Due to the huge computation needed for processing the 3D point cloud, reducing the 3D point cloud into 2D projection is a common way. Moreover, this kind of projection can be fused with camera images as multi-model-based algorithms.

Cristiano [21] proposed Bilateral Filter based upsampling to match the frontview of point cloud projection with related images. Different filter parameters are analyzed to match the field of view of both sensors (LiDAR and camera). Based on this work, different multiview [14,18,20,22] CNN models are proposed for related on-road obstacle detection, sense reconstruction [17,38] and segmentation [5,36] tasks for self-driving.

Xiaozhi Chen et al. [4] also proposed a multiview 3D object detection network that fused different fields of both the LiDAR projection maps and the camera image as input. In their model, both LiDAR-based frontview and BEV projections are used to generate region proposal along with the images.

Complex-YOLO [28] presented an Euler-Region-Proposal based on BEV projection of 3D LiDAR point cloud. This net can predict not only the bounding box of the obstacle, but also the heading angle with the 3D bounding box reconversion module. Simulation on the KITTI dataset shows that this model can achieve real-time detection at 50 frames per second (FPS) on an NVidia Titan X.

Most of the current LiDAR-based detection models are designed based on the dataset of KITTI and nuScenes with a single LiDAR mounted on the vehicle roof, this kind of setup is only suitable for small size vehicles. However, the container trucks are very large, the vehicle head and body will block most of the laser beams, and lead to blind spot. The mounting of the sensors has a great influence on the distribution of the point cloud; therefore, the deep learning model constructed based on the data collected by different solutions cannot be used on other systems. Moreover, the above mentioned models are only designed for high-performance GPU server and do not tested on vehicle environments.

3 Dual-LiDAR Perceptive System

However, as we mentioned above, most of current research works that utilize deep learning model [25] to process 3D point cloud are based on the KITTI dataset. This means that those works can only work with dense LiDAR such as Velodyne HDL-64E. This LiDAR sensor can offer 360° of 64 scan lines of the surrounding environment; however, the price of the sensor is too high. When using low-dense LiDAR such as Velodyne VLP-16 that can only offer 16 laser scan lines, the sampling information may drop to 1/4 of that of the Velodyne HDL-64E. Figure 1 shows the difference between our system and KITTI dataset. This means that we can get less stable sensing data for the later process. That is a big challenge for on-road obstacle detection and tracking for self-driving. Recent works rarely consider the sparse sampling of low-cost LiDAR and vehicle environment deployment as we do.

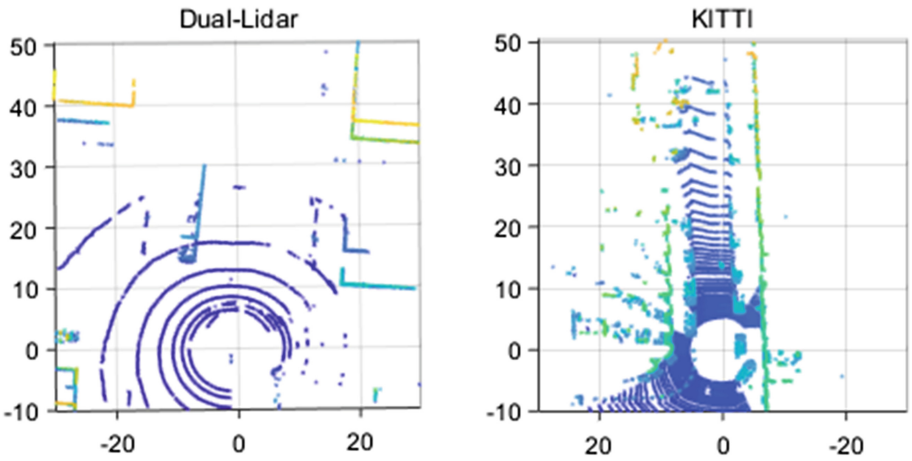


Fig. 1. The left figure shows a BEV LiDAR point cloud projection based on our system, and right one is based on the Velodyne HDL-64E of KITTI dataset.

Figure 2 shows the perception module of our sensor solution for the self-driving container truck. It can be seen that the two low density LiDARs are equipped on the corners of the vehicle head. Both of the two LiDARs' laser scan lines can reach the front area and can be combined as a 32 scan line. For the side areas, each LiDAR covers one side. This sensor deployment simulates the attention mechanism of human drivers.

Although our system has lower sensing performance than dense LiDAR, we investigate several refined modules together to enhance the raw point cloud feature. The vehicle has a maximum speed of 30 km/h in the seaport area. And most vehicles are container trucks with huge size, making them easier to be detected and tracked.

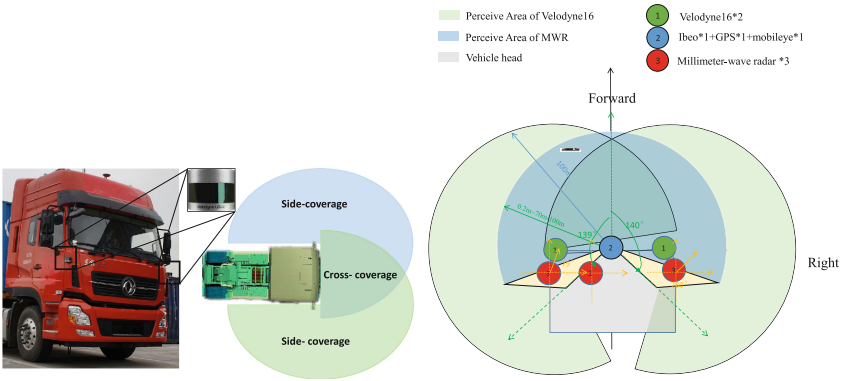


Fig. 2. Two low density LiDARs are equipped on the opposite side of the container truck head (left). The front sensing field (cross-coverage area) is covered by both LiDARs and can achieve double density, and the side-coverage areas are covered by one LiDAR (right). The bottom one is a full sensor scheme used in our self-driving truck.

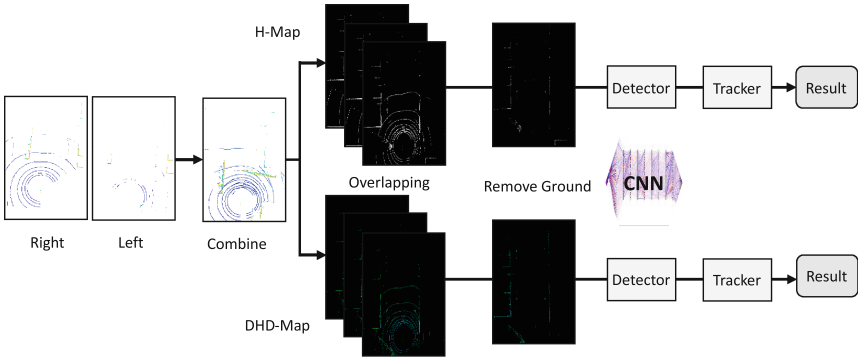


Fig. 3. Workflow of detection and tracking system for our dual-LiDAR container truck

Figure 3 illustrates the whole workflow of dual-LiDAR container truck detection and tracking system. We first calibrate the two LiDAR into the same vehicle body coordinate (with the RTK-GPS as the origin point). For LiDAR BEV projection, two kinds of maps are generated. The H-Map only considers the Height (z-axis) of each point, and the final H-Map contains only one channel (Gray). The HDD-Map considers the Height (z-axis), Depth (y-axis), and Density [39] values of point cloud as a three-channel-map. Equation 1 is the calculation of Density value at each point of i, where C_i is the total points in the local cell. And then, we merge each three successive frames according to the GPS into one to enhance the point cloud features. As the LiDAR sample frequency 10 Hz and installed as in Fig. 2, three-overlapped-frame for the side-view can achieve 48 scan lines, and for front-view can achieve 96 scan lines. We also propose a

simple voxel-height-based algorithm to remove the scan lines on road surfaces. This also leads to clearer obstacle features for later detection.

$$Dense(i) = min(1.0, log\frac{C_i + 1}{64}) \tag{1}$$

With the above pre-processing, the final BEV map will be fed into a lightweight CNN detection model and trained in an 'end-to-end' fashion with carefully labeled bounding boxes. A boosting tracker [10] is used for tracking the obstacles in each frame. We update the tracking items with the CNN detector each 0.5 s to reduce the time consumption of convolution.

3.1 Data Collection and BEV Map Projection

All the data used for model training and testing are collected from a seaport area with our self-driving trucks. Totally, there are three long sequences collected on both sunny and rainy days. The two Velodyne VLP-16 are equipped at the vehicle head of 1.7 m height, therefore, we filter the raw LiDAR point cloud with the following distances: $x \in [-30\,\text{m}, 30\,\text{m}]$, $y \in [-10\,\text{m}, 50\,\text{m}]$ and $z \in [-1.7\,\text{m}, 2.8\,\text{m}]$. In this way, each frame of the point cloud contains a 60 * 60 square with 4.5 m height. This height limitation can cover all the other trucks. Each frame of point cloud generates a 480 * 480 pixel image; therefore, 1 m equals 8 pixels in the image. Different from previous works that do not pay attention to the back view of the vehicle, our ego vehicle is in the back center of the BEV map. For safety reasons, about 10 m of the vehicle tailstock are also considered, because the container truck is nearly 7 m long.

For each long sequence, only the frame that contains obstacles is selected as the short sequence. Therefore, most of the short sequences contain 1 to 8 obstacles. And then, we label the BEV frames, and check the bounding boxes with several different people. The final dataset contains similar sequences of 4647 (sunny), 4594 (cloudy and small rain) and 3973 (sunny) frames, with each having two kinds of projection maps (based on BEV) and well labelled bounding boxes.

3.2 Lightweight CNN Detector

We combine a well designed lightweight convolutional neural network detector with the commonly used multi-target tracker to solve the obstacle tracking task for our self-driving vehicles. And the boosting tracker is used as the basic tracker in the multi-target tracking framework.

Considering the detection performance highly affects the later tracking, the key requirement is to detect all the obstacles efficiently. We combine a resized StridedNet [30] front-bone with the Yolo detection layer as the Strided-Yolo model. Figure 4 and Table 2 are the detailed CNN architecture of our detector model. Large kernel size of $7 * 7$ convolutional layer is first used, followed by small kernel of $3 * 3$ with a stride of 2 that can reduce the original image into 1/2 of the original size. Alternate stacking of $3 * 3$ and $1 * 1$ convolutional layers with different filters at 3–17 layers are used to extract the rich features. At the

second last layer, we up-sample the map and combine it with layer 8. In this way, we use layers 18–23 to extract a two-layer-pyramid-feature at two different scales for Yolo layer.

For tracking stage, we eliminates the point cloud beyond the bounding boxes generated from the detection model. And then, the point cloud containing only the targets uses the same BEV projection to generate the feature maps. At last, the boosting tracker from Opencv is adopted for target matching of consecutive frames.

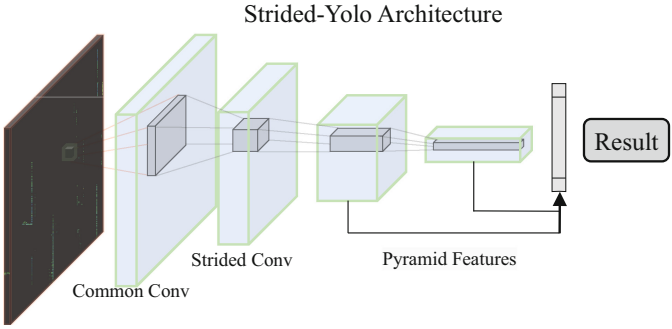


Fig. 4. The architecture of Strided-Yolo model.

Table 2. The detail architecture of Strided-Yolo model.

#	Layer	Parameter	Other OP
1	Conv	7 * 7 * 64/2	Maxpool (2 * 2/1)
2	Conv	3 * 3 * 128/2	Maxpool (2 * 2/1)
3	Conv	1 * 1 * 128/1	Maxpool (2 * 2/1)
4	Conv	3 * 3 * 256/2	Maxpool (2 * 2/1)
5	Conv	1 * 1 * 128/1	Maxpool (2 * 2/1)
6	Conv	3 * 3 * 256/1	Maxpool (2 * 2/1)
7	Conv	1 * 1 * 128/1	Maxpool (2 * 2/1)
8	Conv	3 * 3 * 512/2	Maxpool(2 * 2/1)
9	Conv	1 * 1 * 256/1	*4
10	Conv	3 * 3 * 512/1	
11	Conv	1 * 1 * 256/1	
18	Conv	3 * 3 * 1024/2	
19	Yolo		
20	Combine 8 + 18		
21	Yolo		

4 Experimental Study

4.1 Data Augmentation

Different from version-based image detection, the projection of LiDAR point cloud has more stable features than images. The commonly use projection algorithms mentioned in Sect. 3 use the distance values for sampling. And this kind of data will not be affected by the lighting conditions of the environment. Moreover, the commonly used data augmentation methods, such scale, flip and Gaussian Noise, also have very tiny effects during training. Therefore, we first test our dataset on the benchmark model the, tiny version of Yolo and Yolo-3l [26], on two kinds of projection maps (H-map and HDD-Map). Sequences 1 and 2 are used for training, and Sequence 3 is used for testing. For the random mode, we deployed the above mentioned augmentation methods on the dataset and use only the raw dataset for the non-random mode.

Table 3. Comparison of the training process with and without randomize for LiDAR projection data.

Model		Yolov3-tiny	Yolov3-tiny	Yolov3-tiny-3l	Yolov3-tiny-3l
Modality		H-Map	HDD-Map	H-Map	HDD-Map
FPS		**90.3**	84.5	79.5	75.0
BFLOPs		**7.12**	7.25	9.32	9.45
Random	RPs/Img	20.15	20.63	**14.7**	15.84
	IOU	60.17%	59.53%	59.61%	**63.52%**
	Recall	77.85%	77.26%	79.32%	**85.68%**
Non-random	RPs/Img	16.66	16.08	13.56	**13.19**
	IOU	60.27%	61.60%	62.83%	**64.22%**
	Recall	80.43%	82.5%	84.85%	**85.96%**

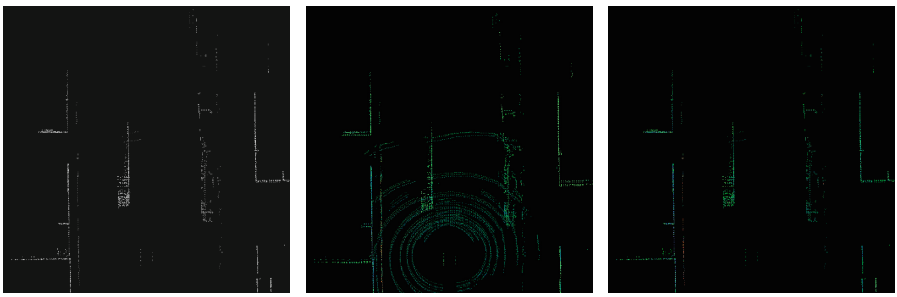
Table 3 illustrates the RPs per image, IOU and Recall of the two models trained and tested on our dataset. For H-Map based models, each frame contains only one channel of LiDAR BEV map, the FPS (Frame Per Second) and model sizes (Billion Float Operations Per Second, BFLOP) are slightly faster and smaller than the HDD-Map (three channels) based models, respectively. It can be seen easy that, the models achieve larger IOU and Recall when training without data augmentation methods. The Recall of Yolov3-tiny trained on H-Map increases by nearly 10% when shut down the random mode in training phase. However, this kind of difference is not so significant for Yolov3-tiny-3l models trained on HDD-Map. The RPs/image is the region proposal detected by the model per image, lower value means the model can reach similar or higher detection accuracy by generating less proposals. This means that when dealing

with LiDAR point cloud projection maps, there is no need to use the traditional data augmentation method. The later training also shuts down the data augmentation function.

The H-Map contains only one channel of the LiDAR point cloud projection, and the models trained on the HDD-Map performs slightly better than the H-Map based models. However, the improvement of HDD-Map is still not very remarkable for it has triple scales of information. This also means that there is much redundant information in the HDD-Map.

4.2 Comparison of Models

We evaluate our model and compared with several state-of-the-arts CNN architectures. Three different kinds of enhanced LiDAR projection map are used: 1) $c1f3g1$: H-Map with the combination of three successive frames and elimination of the ground; 2) $c3f3g0$: HDD-Map with the combination of three successive frames and without elimination of the ground; 3) $c3f3g1$: HDD-Map with the combination of three successive frames and elimination of the ground. Figure 5 illustrates the related projection maps. It can be seen from Fig. 5(b) that simply combining three successive frames can increase the features of the obstacle; however, the scan lines on the ground also lead to noise. In Fig. 5(a) and Fig. 5(c), an H-Map and an HDD-Map with elimination of the ground are generated. Without the ground, the vehicle features are clearer and may help improve the later detection.

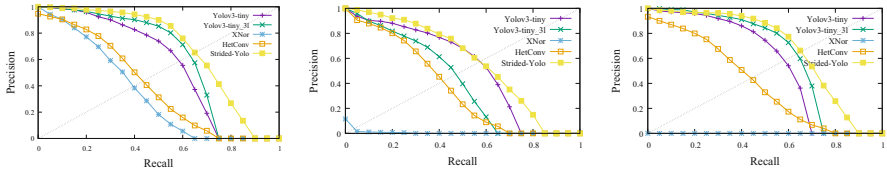


(a) c1f3g1 (b) c3f3g0 (c) c3f3g1

Fig. 5. Difference between three different projection maps with a combination of three frames and elimination of the ground. ($c = channels$, $f = frames$, $g = ground$. PS. $c1f3g1 = 1$ channel of the projection map)

Along with the Strided-Yolo model, we also train 4 different detection models (i.e., Yolov3-tiny, Yolov3-tiny-3l, XNor [24], and HetConv [29]) on the three enhanced LiDAR projection maps (i.e., c1f3g1, c3f3g0, and c3f3g1) for comparison. Figure 6 illustrates the Precision-Recall curves of the five models on the three kinds of projection maps. The point where the line meets the curve is the

break event point (BEP), where $precision = recall$. It can be seen easily that the PR curve of Strided-Yolo can cover all the other curves in Fig. 6(a) and Fig. 6(c). This means that our Strided-Yolo preforms the best on the two kinds of LiDAR projection maps. In Fig. 6(b), the Strided-Yolo and Yolov3-tiny show very similar values at the BEP. The XNor model performs the worst in all the cases, and cannot even converge when dealing with stacked point cloud.



(a) c1f3g1 (b) c3f3g0 (c) c3f3g1

Fig. 6. Precision-Recall Curves of different CNN models with the three projection maps.

Figure 7 shows the PR curves of Strided-Yolo trained on the three stacked projection maps (i.e., c1f3g1, c3f3g0, and c3f3g1) and single frame maps (c1f1g0 and c3f1g0). It can be seen that with more frames, the model can achieve better performance. By eliminating the ground, the models (c1f3g1 and c3f3g1) can achieve higher performance. The models trained on HDD-Map (c3f3g1) and H-Map (c1f3g1) show very similar RP curves.

Table 4. Mean average precision (mAP) of the compared models under different IOU thresholds.

Map	IOU	Model				
		Yolov3-Tiny	Yolov3-Tiny_3l	XNor	HetConv	Strided-Yolo
c1f3g1	0.35	67.85%	71.21%	44.57%	46.90%	73.20%
	0.5	55.96%	60.70%	35.35%	40.14%	66.87%
	0.75	10.81%	10.97%	9.30%	5.17%	31.01%
c3f3g0	0.35	66.98%	62.64%	1.62%	44.45%	64.37%
	0.5	52.75%	41.78%	1.22%	37.37%	57.10%
	0.75	3.75%	6.06%	0.76%	4.91%	19.89%
c3f3g1	0.35	68.03%	71.16%	0.02%	46.06%	70.16%
	0.5	54.50%	61.50%	0.00%	39.60%	66.37%
	0.75	11.48%	9.37%	0.00%	5.02%	29.88%

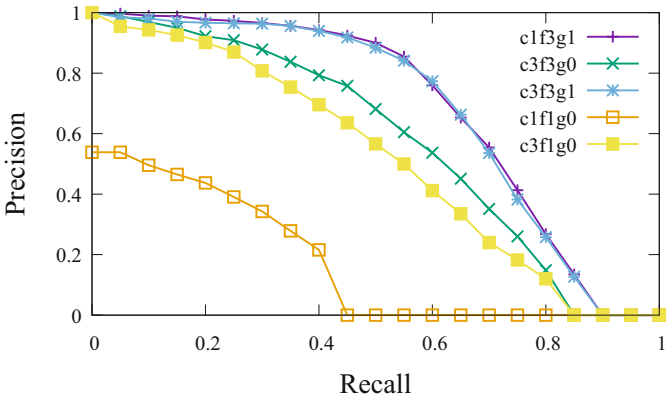


Fig. 7. PR curves of Strided-Yolo on the three projection maps vs. single frame maps.

In Table 4, the mean average precision (mAP) of each model at different intersections over the union (IOU) and LiDAR projection maps is listed. Three kinds of IOU (i.e., 0.35, 0.5 and 0.75) are calculated, and it can be seen that the Strided-Yolo model can achieve the highest mAP in most occasions. Even for $IOU = 0.75$, the Strided-Yolo can still remain about 30% of mAP when training with c1f3g1 and c3f3g1 projection maps, and the value is much higher than the rest of other models. This also means that our model is robust enough for real road conditions. For small IOU (0.35), the Strided-Yolo only performs slightly worse than Yolov3-Tiny and Yolov3-Tiny-3l when using c3f3g0 and c3f3g1, respectively.

Traditional LiDAR detection methods, such as the commonly used DBSCAN clustering [33], are not considered, because they are highly depend on manual tuning, and the robustness of the parameters is very poor. As we tested, the DBScan cannot learn the point cloud distribution of the target, and only relies on the point cloud distance features, which will introduce a large number of false detection results (such as stockade, container, portal crane, etc.), or detect the car head and body as two independent targets. Another disadvantage of the point cloud clustering algorithms is that they cannot provide an accurate target contour, and therefore cannot be used to predict the target size. This makes the quantitative evaluation of traditional algorithms very difficult.

Table 5. Time complexity of the final model with different input projection maps on Tx2.

Map	Preprocess	Detection	Tracking	Total
c1f3g1	0.105 s	0.008 s	0.0046 s	0.1176 s
c3f3g0	0.162 s	0.009 s	0.0046 s	0.1756 s
c3f3g1	0.108 s	0.009 s	0.0046 s	0.1216 s

4.3 Tracking Test

Although the stacked point cloud projection map of c1f3g1 has only one channel of the feature (with the elimination of the ground), it can achieve the highest mAP for Strided-Yolo. Therefore, we use it as our BEV projection map.

The detection and tracking are based CUDA and CNN model, and the processing speed is obtained based on the mean value of the point cloud for 500 consecutive frames, hence the speed. Therefore, the speed is an order of magnitude higher than Arm cpu-based preprocessing. Moreover, the number of dynamic targets in a single frame in the port area is generally about ten, and the matching speed of the boosting tracker is almost negligible. We stack each 3 successive frames of point cloud, and the average processing time for stacking and eliminating the round on Nvidia Jetson Tx2 is listed in Table 5. Most time consumption spends on pre-processing, and the detection and tracking time can be ignored. Moreover, as the Strided-Yolo can achieve very similar performance on the HDD-Map of c3f3g1 and H-Map of c1f3g1, their computational delays are also very close. We choose the c1f3g1 as our final projection map for it has better mAP on all IOU, as shown in Table 4. As the pre-processing module consume a large processing time of 0.1 s, we try to update the detection results for the tracker each 0.5 s. And the tracker will keep tracking for the next 5 frames (i.e., 0.5 s). This means that we stack each 3 continuous point cloud frames for detection, and use the next 5 single frames for tracking.

We also conduct an on-road test with our dual-LiDAR perceptive system on the Nvidia Jetson Tx2 along with other systems on a fully self-driving container truck. The total tracking accuracy can achieve about 87% (with IOU threshold as 0.35), and errors occur mostly when other trucks are driving out of the predefined sensing field of the ego-vehicle.

5 Conclusion

This paper has presented a novel deep learning-based vehicle detection system with two low-cost LiDAR for container trucks in the seaport area. Compare to other commercial solutions presented in Table 1, our sensor cost for low-speed truck is more than half lower than Tusimple. We employ the traditional LiDAR point cloud method to enhance the point cloud by merging continuous frames and eliminating the ground. This will help to maintain clearer vehicle features in BEV projection maps. Different kinds of projection maps are trained with a well-designed CNN model. A comparison with state-of-the-art models shows that our model can achieve the best performance on most occasions. We also test our system on the low-power embedded module of Tx2 with our self-driving system.

Acknowledgment. This work has been supported by China Postdoctoral Science Foundation (2020M681798), Qianjiang Excellent Post-Doctoral Program (2020Y4A001) and 2020 Zhejiang Postdoctoral Research Project (ZJ2020011). JITRI Suzhou Automotive Research Institute Project (CEC20190404). Chongqing Autonomous Unmanned

System Development Foundation and Key Technology Strategic Research Project (2020-XZ-CQ-3). The authors would like to thank Plusgo for their cooperation during data collection.

References

1. Bai, M., Mattyus, G., Homayounfar, N., Wang, S., Lakshmikanth, S.K., Urtasun, R.: Deep multi-sensor lane detection. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3102–3109. IEEE (2018)
2. Caesar, H., et al.: nuScenes: a multimodal dataset for autonomous driving. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11621–11631 (2020)
3. Chang, M.F., et al.: Argoverse: 3D tracking and forecasting with rich maps. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8748–8757 (2019)
4. Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-view 3D object detection network for autonomous driving. In: Computer Vision and Pattern Recognition, pp. 6526–6534. IEEE (2017)
5. Dong, X., Niu, J., Cui, J., Fu, Z., Ouyang, Z.: Fast segmentation-based object tracking model for autonomous vehicles. In: Qiu, M. (ed.) ICA3PP 2020. LNCS, vol. 12453, pp. 259–273. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-60239-0_18
6. Gao, J., Yan, W., Yin, S., Tian, D., Xing, L.: Research on the applicability of automated driving vehicle on the expressway system. Technical report, SAE Technical Paper (2020)
7. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: the KITTI dataset. Int. J. Robot. Res. **32**(11), 1231–1237 (2013)
8. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The KITTI vision benchmark suite. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3354–3361. IEEE (2012)
9. Geyer, J., et al.: A2D2: Audi autonomous driving dataset. arXiv preprint arXiv:2004.06320 (2020)
10. Grabner, H., Grabner, M., Bischof, H.: Real-time tracking via on-line boosting. In: Proceedings of the British Machine Vision Conference (BMVC), vol. 1, pp. 1409–1422 (2006)
11. Le, T., Duan, Y.: PointGrid: a deep network for 3D shape understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 9204–9214 (2018)
12. Li, B.: 3D fully convolutional network for vehicle detection in point cloud. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1513–1518. IEEE (2017)
13. Li, X., Guivant, J.E., Kwok, N., Xu, Y.: 3D backbone network for 3D object detection. arXiv:1901.08373 (2019)
14. Li, Y., Niu, J., Ouyang, Z.: Fusion strategy of multi-sensor based object detection for self-driving vehicles. In: 2020 International Wireless Communications and Mobile Computing (IWCMC), pp. 1549–1554. IEEE (2020)
15. Sun, P., et al.: Scalability in Perception for Autonomous Driving: Waymo Open Dataset. arXiv:1912.04838 (2019)

16. Mokhtar, B., Azab, M., Fathalla, E., Ghourab, E.M., Magdy, M., Eltoweissy, M.: Reliable collaborative semi-infrastructure vehicle-to-vehicle communication for local file sharing. In: Wang, X., Gao, H., Iqbal, M., Min, G. (eds.) CollaborateCom 2019. LNICST, vol. 292, pp. 698–711. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30146-0_47
17. Ouyang, Z., et al.: A cGANs-based scene reconstruction model using lidar point cloud. In: IEEE International Symposium on Parallel and Distributed Processing with Applications. IEEE (2017)
18. Ouyang, Z., Cui, J., Dong, X., Li, Y., Niu, J.: SaccadeFork: a lightweight multi-sensor fusion-based target detector. Inf. Fusion **1**, 1 (2021)
19. Ouyang, Z., Niu, J., Liu, Y., Guizani, M.: Deep CNN-based real-time traffic light detector for self-driving vehicles. IEEE Trans. Mob. Comput. **19**(2), 300–313 (2019)
20. Ouyang, Z., Wang, C., Liu, Yu., Niu, J.: Multiview CNN model for sensor fusion based vehicle detection. In: Hong, R., Cheng, W.-H., Yamasaki, T., Wang, M., Ngo, C.-W. (eds.) PCM 2018. LNCS, vol. 11166, pp. 459–470. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00764-5_42
21. Premebida, C., Garrote, L., Asvadi, A., Ribeiro, A.P., Nunes, U.: High-resolution lidar-based depth mapping using bilateral filter. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), pp. 2469–2474. IEEE (2016)
22. Qi, C.R., Su, H., Nießner, M., Dai, A., Yan, M., Guibas, L.J.: Volumetric and multi-view CNNs for object classification on 3D data. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5648–5656 (2016)
23. Qin, K., Wang, B., Zhang, H., Ma, W., Yan, M., Wang, X.: Research on application and testing of autonomous driving in ports. Technical report, SAE Technical Paper (2020)
24. Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: XNOR-Net: ImageNet classification using binary convolutional neural networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 525–542. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_32
25. Redmon, J., Farhadi, A.: YOLO9000: better, faster, stronger. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition. IEEE (2017)
26. Redmon, J., Farhadi, A.: YOLOv3: an incremental improvement. In: arXiv:1804.02767 (2018)
27. Shi, S., et al.: PV-RCNN: point-voxel feature set abstraction for 3D object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10529–10538 (2020)
28. Simon, M., Milz, S., Amende, K., Gross, H.-M.: Complex-YOLO: an Euler-Region-Proposal for real-time 3D object detection on point clouds. In: Leal-Taixé, L., Roth, S. (eds.) ECCV 2018. LNCS, vol. 11129, pp. 197–209. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11009-3_11
29. Singh, P., Verma, V.K., Rai, P., Namboodiri, V.P.: HetConv: heterogeneous kernel-based convolutions for deep CNNs. In: Computer Vision and Pattern Recognition (CVPR) 2019. IEEE (2019)
30. Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.: Striving for simplicity: the all convolutional net. arXiv preprint (2014)
31. Tang, X., Geng, Z., Chen, W.: Safety message propagation using vehicle-infrastructure cooperation in urban vehicular networks. In: Gao, H., Wang, X., Yin, Y., Iqbal, M. (eds.) CollaborateCom 2018. LNICST, vol. 268, pp. 235–251. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-12981-1_17

32. Verghese, S.: Self-driving cars and lidar. In: CLEO: Applications and Technology, pp. AM3A-1. Optical Society of America (2017)
33. Wang, C., Ji, M., Wang, J., Wen, W., Li, T., Sun, Y.: An improved DBSCAN method for LiDAR data segmentation with automatic Eps estimation. Sensors **19**(1), 172 (2019)
34. Wang, P.S., Liu, Y., Guo, Y.X., Sun, C.Y., Tong, X.: O-CNN: octree-based convolutional neural networks for 3D shape analysis. ACM Trans. Graph. (TOG) **36**(4), 1–11 (2017)
35. Wang, X., Zhang, M., Meng, X., Xia, H., Wu, C., Luo, W.: Development conception and promotion strategy of bus system of the future. In: Proceedings of the 2020 5th International Conference on Cloud Computing and Internet of Things, pp. 63–68 (2020)
36. Wu, B., Li, P., Chen, J., Li, Y., Fan, Y.: 3D environment detection using multi-view color images and lidar point clouds. In: 2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW), pp. 1–2. IEEE (2018)
37. Yang, B., Luo, W., Urtasun, R.: PIXOR: real-time 3D object detection from point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7652–7660 (2018)
38. Yi, C., et al.: Urban building reconstruction from raw LiDAR point data. Comput. Aided Des. **93**, 1–14 (2017)
39. Zhao, J., Zhang, X.N., Gao, H., Yin, J., Zhou, M., Tan, C.: Object detection based on hierarchical multi-view proposal network for autonomous driving. In: 2018 International Joint Conference on Neural Networks (IJCNN), pp. 1–7. IEEE (2018)
40. Zhou, Y., Tuzel, O.: VoxelNet: end-to-end learning for point cloud based 3D object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4490–4499 (2018)