https://doi.org/10.26599/AIR.2022.9150009 CAAI Artificial Intelligence Research

LWD-3D: Lightweight Detector Based on Self-Attention for 3D Object Detection

Shuo Yang[1], Huimin Lu[1,2] ✉, Tohru Kamiya[1], Yoshihisa Nakatoh[1], and Seiichi Serikawa[1]

ABSTRACT

Lightweight modules play a key role in 3D object detection tasks for autonomous driving, which are necessary for the application of 3D object detectors. At present, research still focuses on constructing complex models and calculations to improve the detection precision at the expense of the running rate. However, building a lightweight model to learn the global features from point cloud data for 3D object detection is a significant problem. In this paper, we focus on combining convolutional neural networks with self-attention-based vision transformers to realize lightweight and high-speed computing for 3D object detection. We propose light-weight detection 3D (LWD-3D), which is a point cloud conversion and lightweight vision transformer for autonomous driving. LWD-3D utilizes a one-shot regression framework in 2D space and generates a 3D object bounding box from point cloud data, which provides a new feature representation method based on a vision transformer for 3D detection applications. The results of experiment on the KITTI 3D dataset show that LWD-3D achieves real-time detection (time per image < 20 ms). LWD-3D obtains a mean average precision (mAP) 75% higher than that of another 3D real-time detector with half the number of parameters. Our research extends the application of visual transformers to 3D object detection tasks.

KEYWORDS

3D object detection; point clouds; vision transformer; one-shot regression; real-time

The realization of autonomous driving requires an efficient environment perception system to complete 3D object detection to achieve the goal of avoiding obstacles and road traffic. LiDAR sensors play a key role in the data input for environment perception systems. They are also essential components for 3D object detection and recognition. Point cloud data are an important output format of LiDAR sensors, which have the physical ability to perceive depth in 3D space. Thus, extracting object features from point clouds to explore 3D object detection is a significant research direction.

In contrast to RGB images, point cloud data consist of points based on a 3D coordinate system as object descriptors[1], which include depth and position information. Figure 1 shows an example of an RGB image with point cloud data. Three-dimensional object detection based on autonomous driving is a typical research topic for exploring point clouds. However, the sparsity and disorder of point cloud data affect the model construction for 3D object detectors, which influences the processing speed. Previous work focused on improving the accuracy of 3D object detection based on deep learning and ignored its application in practice. We found that building a lightweight module is conducive to the application of 3D object detection tasks in the field of autonomous driving.

According to a survey of 3D object detection methods, three typical methods are available for feature representation from point cloud data, namely, multi-view-based methods, voxel-based methods, and point-based methods. Multi-view-based methods build an orientation view from the point cloud to represent the object features. Then, they use 2D convolutional neural networks to extract features and predict the bounding box. Their advantage

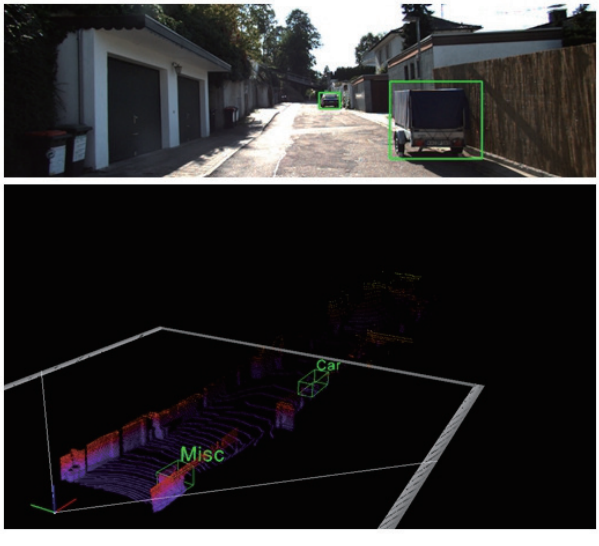


Fig. 1 Examples of RGB images and 3D point cloud data for bounding boxes in autonomous driving scenes.

lies in their fast calculation speed. However, the model design does not adjust to the 3D object, which causes the loss of detailed features and affects the accuracy. Voxel-based methods convert the discrete and unordered point cloud into a regular matrix in a limited 3D space by calculating the spatial density of the point cloud. These methods determine the size of the converted matrix according to the number of points in the point cloud. Thus, the feature loss is small compared to that of multi-view-based methods. The disadvantage is that the overall calculation speed is

1 School of Engineering, Kyushu Institute of Technology, Fukuoka 804-8550, Japan
2 School of Information Engineering, Yangzhou University, Yangzhou 225127, China.
Address correspondence to Huimin Lu, luhuimin@ericlab.org

slow and decreases exponentially as the number of points in the point cloud increases. Point-based methods directly address the relationships among point cloud data by using a cascade of multi-layer perceptron (MLP). This effectively guarantees the invariance of the input point cloud and improves the computational efficiency under certain conditions.

However, these three typical calculation methods still cannot meet the speed requirements for 3D object detection in practical autonomous driving applications. According to our analysis, the previous work did not truly consider the lightweight processing of the model or its deployment to devices with limited computing hardware. Thus, most methods still have many problems, such as a large number of intermediate parameters to store, large weights and high computational complexity. We summarize the current important influencing factors for lightweight processing.

(a) Point cloud processing. The numbers of point cloud data acquired by different cameras are different. Improving the data processing efficiency without the loss of point cloud data is important for solving this problem.

(b) Feature representation network design. There are many model structures that can be used for feature extraction, such as 3D sparse convolutional neural networks and PointNet[2]. Reducing the number of redundant parameters and fusing the advantages of the models are essential for solving this problem.

(c) Detection framework design. In previous work, a two-stage detection framework was mainly used, i.e., generating region proposals and completing two classification and regression calculations for bounding box determination, which originated from the theory of 2D object detection. However, this framework severely limits the prediction efficiency of the model.

In this paper, we focus on designing a lightweight feature extraction network and detection framework for applying a 3D object detector to autonomous driving. Meanwhile, we propose a faster and simpler point cloud preprocessing method that does not affect the quality of the input point cloud information. For the detection framework, we further reduce the detection calculation process to decrease the number of redundant calculations and improve the prediction efficiency. The contributions of our module are summarized as follows:

(a) We propose a conversion method from point clouds by 3D-to-2D bird's eye view (BEV) generation, which can record the transformation matrix to preserve the corresponding relationships between point clouds with channel parameters.

(b) We combine lightweight convolutional neural networks with a transformer module to build feature extraction networks, which can balance the detection accuracy and running rate.

(c) We design a one-stage detection framework for 3D object detection based on "you only look once" (YOLO), which uses prediction layers with different scales to achieve a higher detection running rate.

(d) We use the KITTI 3D dataset[3] to evaluate the LWD-3D method. This method achieves real-time detection performance for autonomous driving.

The remainder of this paper is organized as follows: Section 1 introduces the related work for 3D object detection based on point cloud processing. Section 2 describes in detail the module for our LWD-3D method. Section 3 presents the experimental results with the compared method and ablation analysis. Section 4 presents the conclusions and discusses future research.

1 Related Work

In our work, we contribute to the exploration of real-time detection methods for autonomous driving systems. To summarize the previous work, we analyze 3D object detection methods based on deep learning. According to our survey, 4 types of 3D object detection methods are available, namely, multi-view-based methods, voxel-based methods, point-based methods, and real-time methods.

1.1 Multi-view-based methods

In previous work, most methods cannot use a point cloud as the object of data processing. Multiview-based methods build a feature extraction network by converting the point cloud data into multi-view images[4,5]. For example, in the typical method, a learning network for BEV images is proposed by combining the RGB images. This method uses the data augmentation method for 3D detection by generating RGB images from different angles. Then, MV3D[6] improves the detection performance by building a region proposal network based on a two-stage detection framework, which only uses BEV images to predict the detection result from point clouds. Recently, a stereo-based automatic alignment learning method[7] was proposed for left and right images, which uses point matching to optimize the bounding box to improve the detection performance. This method combines global features with local features from BEV images from different angles by converting the point clouds. However, the conversion method causes the loss of detailed features during the process of converting the point clouds to BEV images, which influences the detection precision.

1.2 Voxel-based methods

Recently, voxel-based methods have played a key role in the field of 3D object detection and segmentation and have achieved good performance in feature extraction. The typical methods are VoxelNet[8] and VoxNet[9]. These methods utilize a feature extraction module based on a 3D sparse convolutional neural network, which uses the voxelization method to process a point cloud. In these methods, every voxel grid needs to calculate the number of point cloud data in the self-regions. For feature extraction, these methods can provide detailed information from point clouds and decrease the loss during conversion processing. Thus, most related works[10,11] have focused on detection frameworks based on voxel-based methods. However, these methods have high time cost and cannot realize real-time detection using a two-stage detection framework, such as PointRCNN[12]. Using a one-stage detection framework, the SE-SSD method[13] uses a single regression method with a self-attention network from the point cloud data to improve the detection running rate. This method focuses on improving the feature representation module to increase the detection accuracy, but it still spends calculation time on the self-attention network stage.

1.3 Point-based method

The main strategy of point-based methods is to directly apply the point cloud for feature extraction and object detection. There are two typical methods, PointNet and PointNet++[14], which use MLP to represent the features from point clouds. The advantages of these methods are that they reduce the complexity of the model for 3D object classification, detection, and segmentation. Therefore, most methods[15–17] design the module using the changed point-based method to realize good performance on object perception tasks. However, this method still has the problem that it needs to resize large point clouds into point clouds

with a limited amount of data in the data preprocessing stage. Through the resizing process, detailed information on the 3D space from the point clouds is still lost. In addition, point-based methods do not perform well in the process of global feature extraction.

1.4 Real-time methods

Three-dimensional object detection is a part of the autonomous driving foundation, yet its time cost and computational complexity must be considered. Thus, real-time methods are needed to make the method as efficient as possible in terms of prediction and running time while maintaining good precision. For 3D object detection, previous research has been performed to develop detectors with improved efficiency, such as the YOLO-based 3D detector[18] and SSD-based detector[19]. These methods cannot realize good detection accuracy by reducing the processing due to the sparsity of the point cloud. Most methods are still based on the characteristics of 2D detection data, which may result in incorrect or invalid 3D bounding boxes. Therefore, achieving a more reasonable and efficient application of 3D detection models based on deep learning is still an important research direction of autonomous driving.

In this paper, we contribute to the development of lightweight networks for feature representation from point clouds by applying MobileNet[20] with a transformer module. To reduce the computational complexity, we propose a fast-processing method for point clouds to achieve deployment to devices with limited computing hardware.

In this research, we focus on the feature fusion of point clouds by combining voxelization-based methods with PointNet-based methods to achieve the common representation of global features and local features. To compensate for the loss in the sampling process, we use the self-attention mechanism method and region proposal-based monitoring method to adjust and control the sampling process.

2 Proposed Method

In this work, we develop a lightweight detection module based on deep learning for 3D object detection in the scenario of autonomous driving. Through an analysis of the 2D detection module and 3D detection module, we propose a state-of-the-art method for real-time detection, namely, LWD-3D. This method includes three parts: a point cloud conversion module, mobile transformer feature extraction module, and detection head for 3D transformation. Through the point cloud conversion module, we build a method for converting 2D images into 3D point clouds, which has no memory usage problems with intermediate parameters. The mobile transformer feature extraction module can provide effective global and local features by utilizing a self-attention module. The detection head for 3D transformation builds a more efficient detection framework to further accelerate inference.

2.1 Pipeline of LWD-3D

As shown in Fig. 2, LWD-3D is an end-to-end training and prediction framework for 3D object detection. It includes 3 stages for point cloud processing and result output. The data input uses the method of BEV generation to convert the point cloud, which records the transformation matrix for the conversion function. We propose a transformation function for calculating the point cloud to match the BEV images of each pixel grid. According to the number of point cloud data in the scene, the parameters of the transformation function can be adjusted to address the generalization of the model. In addition, a transformation matrix is used to generate a 3D bounding box from the BEV detection result of the 2D bounding box.

The feature extraction network combines convolution with depth-wise separable convolutional networks to build a feature representation module. To acquire the detailed distance features from the point cloud, we use a transformer block to connect the feature encoder and decoder modules. Depth-wise separable convolutional networks are only one-third of the original convolution computation and can effectively improve the calculation speed and reduce the parameter quantity. Meanwhile, the transformer structure can effectively supplement the distance features between point clouds to compensate for the feature loss caused by depth-wise separable convolution.

For the detection head, we propose a one-stage detection framework based on YOLO[21,22]. Compared with the 2D object detector, our module needs to complete the regression calculation of the 3D bounding box so that the positioning result after combining the transformation matrix is more accurate for 3D detection. To improve the efficiency of detection training, we use the K-means clustering method to preprocess the dataset, which can acquire the initial value of the anchor box. This method is helpful for improving the training efficiency and running speed of the model. After obtaining the 2D bounding box on the BEV images, our model generates a 3D bounding box by combining the transformation matrix from the data input.

In the pipeline of our method, the point cloud conversion module converts and records the input data of the point cloud through the BEV image generation method by using a density calculation function. The mobile transformer feature extraction module extracts the features from the BEV image with the lightweight convolution and transformer. Finally, the detection head predicts the classification and localization results based on the candidate anchor box method. It is significant that our method is an efficient detection framework that can be applied to practical object detection on limited computing devices.

2.2 Point cloud conversion module

To reduce the amount of data in the feature extraction stage, we propose a point cloud conversion method, in which the point cloud is projected to build a BEV image. With MV3D[6] as a reference, we use the multichannel view to represent the point cloud. Thus, we build a grid image from a projection of the point cloud, as shown in Fig. 3. For the unordered point cloud, we generate an $n \times n$ grid cell, and we calculate the point for each grid cell. The left image represents the maximum height calculation for the point cloud, where each grid cell uses the pixel value to represent the height of the highest point with the related cell. Thus, the left image can acquire the space distance information from the point cloud. The formula for the height calculation is

$$H_z = \max(P_{\text{point}} \cdot [0,0,1]^{\mathrm{T}}) \tag{1}$$

where H_z represents the height of the highest point from the 3D data and P_{point} represents the collection of all points in the area.

The right image builds the density feature map for the point cloud. The formula for the density is as follows:

$$D_z = \min\left(1.0, \log\left(\frac{N+1}{64}\right)\right) \tag{2}$$

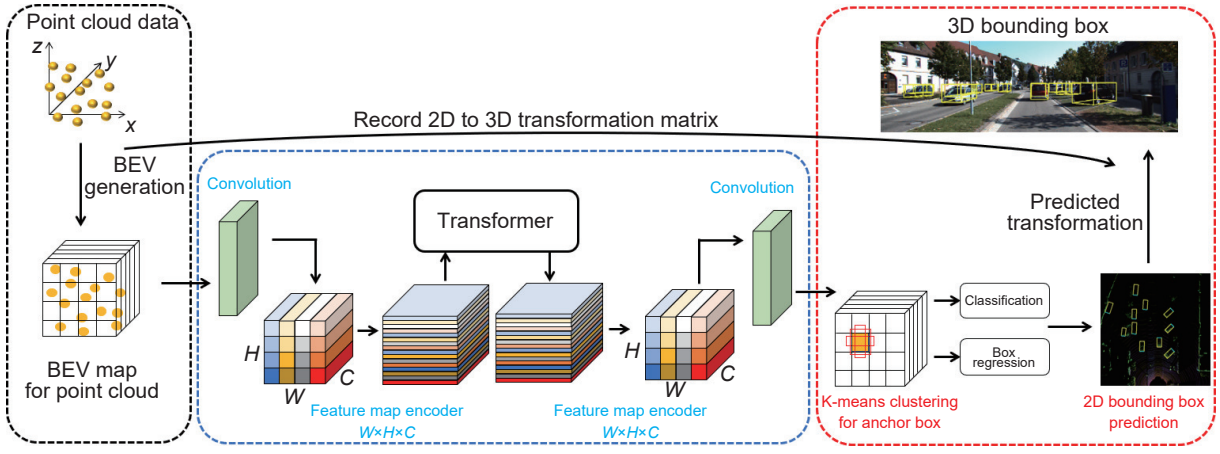
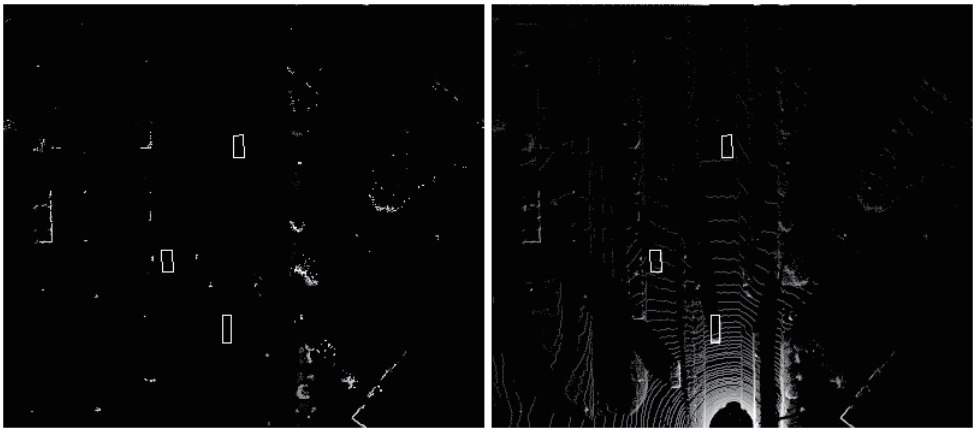


Fig. 2 Example of a 3D transformer down-sampling network. We use 2D images to represent the voxel point cloud data. Among them, gray grids represent the nonempty voxels. Through the self-attention module, we can obtain the confidence score of each nonempty voxel. Therefore, our method builds a more robust object feature representation model in the feature extraction network.



(a) (b)

Fig. 3 Example of converting an image from a point cloud. (a) Height image from the maximum height calculation result for the point cloud. (b) Density image from the result of the density feature map for the point cloud.

where D_z is the density value of the point cloud for each grid cell and N represents the number of points in each grid cell.

In addition, we recode the intensity information of the point cloud as the input to compensate for the loss during the conversion process. The formula for the intensity calculation for the point cloud is

$$I_z = \max\left(I(P_{\text{point}})\right) \tag{3}$$

where I_z is the maximum intensity for each grid cell and I represents the calculation of the intensity.

Thus, we combine the three images to establish a new input format for point cloud processing. This method reduces the amounts of feature processing and overall calculation in the subsequent process for feature extraction.

2.3 Mobile transformer feature extraction module

In this work, we focus on combining convolutional neural networks with visual transformers to represent the local and global features from point clouds. To improve the running rate, we use the depth-wise separable convolution to replace the original convolution based on MobileNet[20].

Figure 4 shows the calculation process for the depth-wise separable convolution operation. The main strategy is to convolve the channel for each input data instance without adding a large number of network kernels. Then, 1×1 kernels are used to expand the channels from the previous layer. Therefore, the overall computational burden of this model is only one-third that of ordinary convolution. In this work, our module comprises 3×3 channel kernels and 1×1 fusion kernels. However, the depth-wise separable convolution cannot provide more features to facilitate detailed local feature extraction.

The transformer module constructs a better relationship learning model between local grid cells. Thus, we want to use the fusion module to model the local and global features for point clouds with fewer parameters and calculations. In previous approaches, the transformer module may lose the order relationship of point cloud information. Our module loses neither the path order nor the spatial order for each point cloud by using the depth-wise separable convolution to recode the feature tensor before the input transformer.

For transformer processing, this model calculates the relationship of each grid cell, which can fuse the concatenated features from the point cloud in each grid. Thus, our model can encode and decode the global features from all pixels of the grid cells. Compared to the previous 3D detection module, our method uses fewer parameters and computations to build the feature extraction network, which has the advantage of a combined representational module. With the convolution and transformer in our method, the resultant feature extraction block has convolution-like properties for representing the global features.

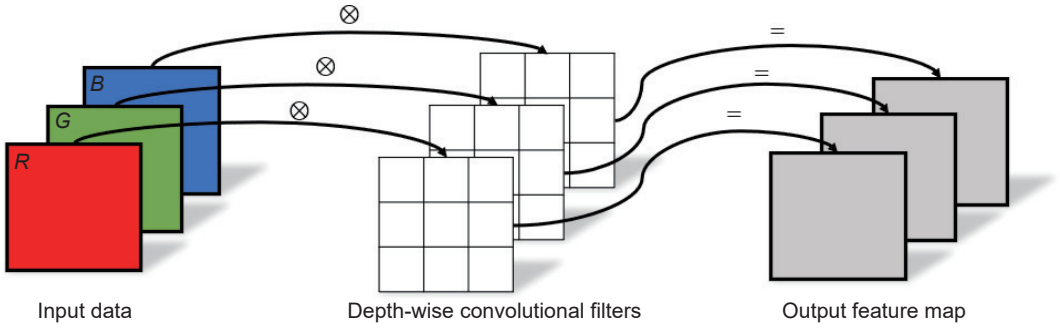


Fig. 4 Structure of depthwise separable convolution.

According to this module, we can decrease the number of network layers, making the network lightweight.

2.4 Detection head

The main strategy of YOLO is to use K-means clustering to set anchor boxes on 2D images. Network fine-tuning requires the priors of each bounding box. Of course, each prior anchor box must be able to cover the whole region of the corresponding object in the dataset. For the 3D detection task, the anchor-based method is beneficial for locating the bounding box of an object, which can train on objects of different scales. Thus, the lightweight detection module needs the anchor-based method to ensure location accuracy, especially for 3D objects.

In this detection head, we build a prior anchor box for the 3D box (x, y, z, w, h, c). For the original anchor box, we add the z coordinate to represent the height of the box. Meanwhile, we combine the transformation matrix from the point cloud conversion to further improve the accuracy of the bounding box regression calculation. In addition, for pedestrians and cyclists, we further broaden the setting range of the a priori bounding box to improve the generalization performance of the overall detection framework.

For the bounding box regression, we still use the upper-left corner of the whole image as the coordinate origin and establish the 3D bounding box regression calculation principle. Figure 5 shows an example of the center coordinate of each grid cell. According to the bounding box method, our detection framework only performs one regression and classification calculation, which effectively improves the calculation efficiency and reduces the number of model parameters. In addition, we implement end-to-end training and prediction methods to meet the requirements for practical application.

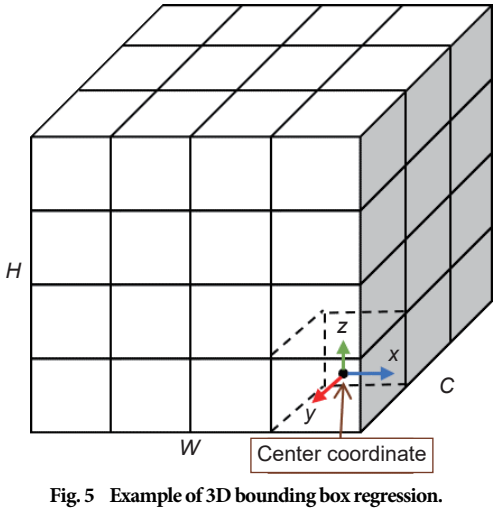


Fig. 5 Example of 3D bounding box regression.

2.5 Training policy

Due to the dispersion and disorder of point clouds, the data conversion process also produces diverse changes in object scale. On the BEV images, the 3D object still has a scale change problem. Meanwhile, the 3D object, e.g., from the pedestrian class, is smaller in the overall images. Therefore, we propose a multistage pretraining policy to improve the training efficiency of the models. The density of the point cloud comes from the results of the transformation stage. We use different scales to train the feature extraction network to adapt to the scale change of the input object. The formula $Scale_{\text{Inputdata}}$ is

$$Scale_{\text{Inputdata}} = 32 \times (13 + n), n \in [0, 7] \tag{4}$$

where 32 represents the down-sampling ratio for feature extraction network. 13 present the minimum output feature image size. Through the adjustment of n, the model training can use different size data for training. For data enhancement, we perform geometric rotation, random clipping and block input in the process of BEV image generation. Compared with the original 3D data, the 2D data make it easier to design data enhancement methods.

3 Experiment

For experimental evaluation, we use the KITTI 3D detection dataset. In the previous work, few methods consider model efficiency or practical applications. Therefore, we select only one type of method for comparison, namely, Complex-3D[18]. In this section, we present in detail the dataset, parameters, experiment and analysis.

3.1 Dataset

KITTI 3D detection dataset is a typical benchmark for 3D object detection tasks in the scenario of autonomous driving. The dataset consists of 7481 training samples with label information and 7518 test samples without labels for the 3D bounding box. Although the dataset contains 10 categories, the categories of this dataset are uneven. Thus, we only select and experiment with the 3 large categories of objects, namely, Car, Pedestrian, and Cyclist.

According to the dataset principles, we evaluate our models at 3 levels of difficulty: easy, moderate, and hard. We divide the dataset into a training dataset (3712 images) and validation dataset (3769 images).

For the processing of the dataset, we project the point cloud into 2D space to obtain a BEV image with a resolution of 0.1 meters per pixel. Meanwhile, we set the range of the LiDAR sensors to 40 meters to the right, 40 meters to the left, and 80 meters forward in 3D space. Using this range, the input images of the BEV map are of pixel size 640×640. For the maximum point

cloud, we limit the spatial height of the point cloud to ±2 meters.

We complete the evaluation experiment on the KITTI 3D detection dataset. Meanwhile, we select a typical and state-of-the-art lightweight 3D detection method for the comparative experiments. For the indicators of evaluation, we use the average precision (AP) and mean average precision (mAP) for multiple classes with a 3D intersection over union (IOU) threshold of 0.3.

3.2 Training and parameters

Our module is trained using an end-to-end method. We use stochastic gradient descent with the following training super parameters: a momentum of 0.9 and a weight decay of 0.0005. We set the threshold to 0.5 for the 3D IOU according to the evaluation principle.

All of our modules are trained and tested in an end-to-end manner on an NVIDIA Tesla A8000 GPU. For the training policy, our training module has an initial learning rate of 0.001. We train our module for 300 epochs with a batch size of 64 GPU cards. We use the data augmentation method on the KITTI 3D dataset to train the BEV images.

3.3 Comparison experiment on the KITTI dataset

For the evaluation, we select the state-of-the-art model for real-time detection as the comparative method, which also uses a one-stage detection framework. The data input only uses the point cloud from the LiDAR sensor to convert the BEV images. Table 1 shows the comparison experimental results for the 3 classes (Car, Pedestrian, and Cyclist). For the small size objects of pedestrian and cyclist, our method still has higher detection performance in qualitative results.

4 Conclusion

In this work, we proposed a lightweight network for 3D object detection for autonomous driving, called LWD-3D, which utilizes a new conversion method from point clouds and feature representation modules. The proposed method includes 3 parts: a point cloud conversion module, mobile transformer feature extraction module, and detection head. First, we focused on simplifying feature processing and memory occupation. Thus, we proposed a conversion method from 3D to 2D BEV images. Then, we constructed a feature fusion module for combining the depth-wise separable convolution using a transformer to represent the local and global features. Finally, we used the one-stage detection framework to predict the 3D bounding box and classes based on the transformation matrix. For evaluation, the experimental results proved the efficiency and precision of LWD-3D on the KITTI 3D detection dataset; thus, our method realized

Table 1 Results of evaluation experiments in AP comparing the two methods on the KITTI dataset.

Stage	Method	Modality	Parameter	AP (%)									Time (ms)
				Car			Cyclist			Pedestrian			
				Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard	
One-stage	Complex-YOLO	LiDAR	4.56×10^7	68.33	42.08	37.45	20.25	19.65	15.37	20.52	19.16	14.28	20
	YOLO-6D	LiDAR	8.6×10^7	70.15	47.29	40.23	24.17	22.58	18.41	23.82	21.63	17.84	35
	LWD-3D	LiDAR	5.6×10^6	75.41	52.15	43.29	28.31	30.25	24.57	29.47	30.28	24.12	8
	Improvement			**+7.08**	**+10.07**	**+5.84**	**+8.06**	**+10.6**	**+9.2**	**+8.95**	**+11.12**	**+9.84**	−12

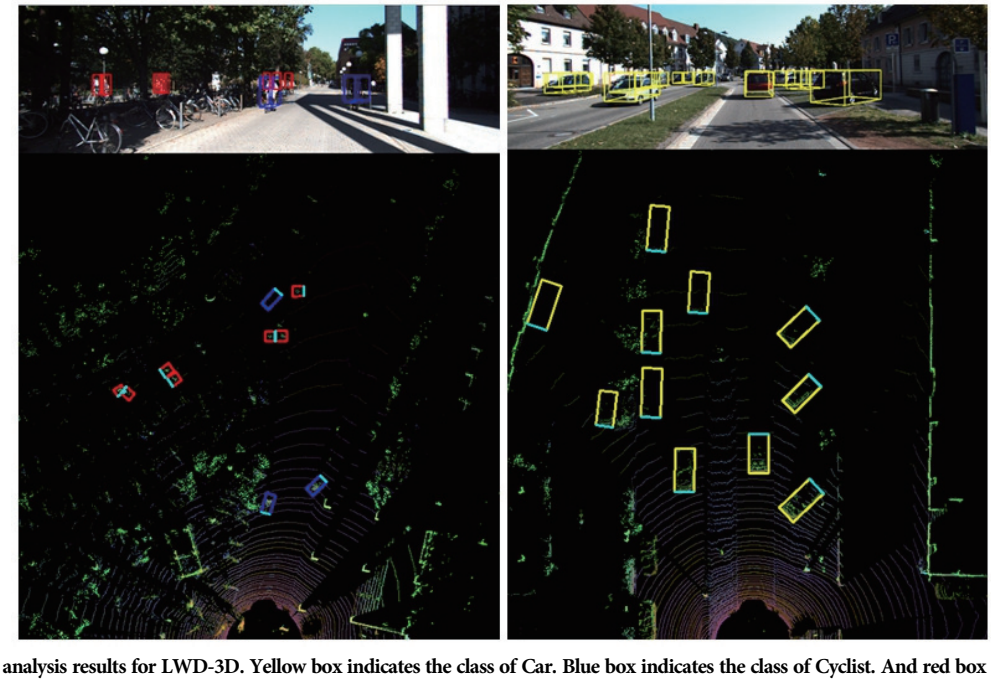


Fig. 6 Qualitative analysis results for LWD-3D. Yellow box indicates the class of Car. Blue box indicates the class of Cyclist. And red box indicates the class of Pedestrian.

state-of-the-art performance on real-time detection. In particular, our proposed method achieved good running rates under various levels of difficulty. In our future research, we plan to concentrate on optimizing the precision for real-time detection by fusing the convolution and transformer modules.

Acknowledgment

This work was partially supported by the National Natural Science Foundation of China (No. 62206237), Japan Science Promotion Society (Nos. 22K12093 and 22K12094), and Japan Science and Technology Agency (No. JPMJST2281).

Article History

Received: 5 December 2022; Revised: 1 January 2023; Accepted: 8 January 2023

References

[1] S. Shi, L. Jiang, J. Deng, Z. Wang, C. Guo, J. Shi, X. Wang, and H. Li, PV-RCNN++: Point-voxel feature set abstraction with local vector representation for 3D object detection, arXiv preprint arXiv: 2102.00463, 2021.

[2] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, PointNet: Deep learning on point sets for 3D classification and segmentation, in *Proc. 2017 IEEE Conf. Computer Vision and Pattern Recognition* (*CVPR*), Honolulu, HI, USA, 2017, pp. 77–85.

[3] A. Geiger, P. Lenz, and R. Urtasun, Are we ready for autonomous driving? The KITTI vision benchmark suite, in *Proc. 2012 IEEE Conf. Computer Vision and Pattern Recognition*, Providence, RI, USA, 2012, pp. 335–336.

[4] M. Liang, B. Yang, Y. Chen, R. Hu, and R. Urtasun, Multi-task multi-sensor fusion for 3D object detection, in *Proc. 2019 IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), Long Beach, CA, USA, 2019, pp. 7337–7345.

[5] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, Joint 3D proposal generation and object detection from view aggregation, in *Proc. 2018 IEEE/RSJ Int. Conf. Intelligent Robots and Systems* (*IROS*), Madrid, Spain, 2018, pp. 1–8.

[6] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, Multi-view 3D object detection network for autonomous driving, in *Proc. 2017 IEEE Conf. Computer Vision and Pattern Recognition* (*CVPR*), Honolulu, HI, USA, 2017, pp. 6526–6534.

[7] P. Li, X. Chen, and S. Shen, Stereo R-CNN based 3D object detection for autonomous driving, in *Proc. 2019 IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), Long Beach, CA, USA, 2019, pp. 7636–7644.

[8] Y. Zhou and O. Tuzel, VoxelNet: End-to-end learning for point cloud based 3D object detection, in *Proc. 2018 IEEE/CVF Conf. Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, pp. 4490–4499.

[9] D. Maturana and S. Scherer, VoxNet: A 3D convolutional neural network for real-time object recognition, in *Proc. 2015 IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, Hamburg, Germany, 2015, pp. 922–928.

[10] X. Chen, K. Kundu, Y. Zhu, H. Ma, S. Fidler, and R. Urtasun, 3D object proposals using stereo imagery for accurate object class detection, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 5, pp. 1259–1272, 2018.

[11] Y. Chen, S. Liu, X. Shen, and J. Jia, Fast point R-CNN, in *Proc. 2019 IEEE/CVF Int. Conf. Computer Vision* (*ICCV*), Seoul, Republic of Korea, 2019, pp. 9774–9783.

[12] S. Shi, X. Wang, and H. Li, PointRCNN: 3D object proposal generation and detection from point cloud, in *Proc. 2019 IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), Long Beach, CA, USA, 2019, pp. 770–779.

[13] W. Zheng, W. Tang, L. Jiang, and C. W. Fu, SE-SSD: Self-ensembling single-stage object detector from point cloud, in *Proc. 2021 IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), Nashville, TN, USA, 2021, pp. 14489–14498.

[14] C. R. Qi, Y. Li, H. Su, and L. J. Guibas, PointNet++: Deep hierarchical feature learning on point sets in a metric space, in *Proc. 31st Int. Conference on Neural Information Processing Systems*, Long Beach, CA, USA, 2017, pp. 5105–5114.

[15] Y. Yan, Y. Mao, and B. Li, SECOND: Sparsely embedded convolutional detection, *Sensors*, vol. 18, no. 10, p. 3337, 2018.

[16] C. R. Qi, O. Litany, K. He, and L. Guibas, Deep Hough voting for 3D object detection in point clouds, in *Proc. 2019 IEEE/CVF Int. Conf. Computer Vision* (*ICCV*), Seoul, Republic of Korea, 2019, pp. 9276–9285.

[17] 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. 2019 IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), Long Beach, CA, USA, 2019, pp. 12689–12697.

[18] M. Simon, S. Milz, K. Amende, and H. M. Gross, Complex-YOLO: Real-time 3D object detection on point clouds, arXiv preprint arXiv: 1803.06199v2, 2018.

[19] Z. Yang, Y. Sun, S. Liu, and J. Jia, 3DSSD: Point-based 3d single stage object detector, in *Proc. 2020 IEEE/CVF Conf. Computer Vision and Pattern Recognition* (*CVPR*), Seattle, WA, USA, 2020, pp. 11037–11045.

[20] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, MobileNets: Efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv: 1704.04861, 2017.

[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, You only look once: Unified, real-time object detection, in *Proc. 2016 IEEE Conf. Computer Vision and Pattern Recognition* (*CVPR*), Las Vegas, NV, USA, 2016, pp. 779–788.

[22] J. Redmon and A. Farhadi. YOLO9000: Better, faster, stronger, in *Proc. 2017 IEEE Conf. Computer Vision and Pattern Recognition* (*CVPR*), Honolulu, HI, USA, 2017, pp. 6517–6525.