

Article

Three-Dimensional Object Detection Network Based on Multi-Layer and Multi-Modal Fusion

Wenming Zhu ^{1,†}, Jia Zhou ^{2,†}, Zizhe Wang ¹, Xuehua Zhou ^{2,3,*}, Feng Zhou ¹, Jingwen Sun ¹, Mingrui Song ² and Zhiguo Zhou ^{2,3} 

¹ Changzhou Power Supply Branch of State Grid Jiangsu Electric Power Co., Ltd., Changzhou 213000, China; zhuwm@js.sgcc.com.cn (W.Z.); wangzz6@js.sgcc.com.cn (Z.W.); zhoul31@js.sgcc.com.cn (F.Z.); sunjw4@js.sgcc.com.cn (J.S.)

² School of Integrated Circuits and Electronics, Beijing Institute of Technology, Beijing 100081, China; 3120210720@bit.edu.cn (J.Z.); 3220231873@bit.edu.cn (M.S.); zhiguo Zhou@bit.edu.cn (Z.Z.)

³ Tangshan Research Institute of BIT, Tangshan 063000, China

* Correspondence: xuehuazhou@bit.edu.cn

† These authors contributed equally to this work.

Abstract: Cameras and LiDAR are important sensors in autonomous driving systems that can provide complementary information to each other. However, most LiDAR-only methods outperform the fusion method on the main benchmark datasets. Current studies attribute the reasons for this to misalignment of views and difficulty in matching heterogeneous features. Specially, using the single-stage fusion method, it is difficult to fully fuse the features of the image and point cloud. In this work, we propose a 3D object detection network based on the multi-layer and multi-modal fusion (3DMMF) method. 3DMMF works by painting and encoding the point cloud in the frustum proposed by the 2D object detection network. Then, the painted point cloud is fed to the LiDAR-only object detection network, which has expanded channels and a self-attention mechanism module. Finally, the camera-LiDAR object candidates fusion for 3D object detection (CLOCs) method is used to match the geometric direction features and category semantic features of the 2D and 3D detection results. Experiments on the KITTI dataset (a public dataset) show that this fusion method has a significant improvement over the baseline of the LiDAR-only method, with an average mAP improvement of 6.3%.



Citation: Zhu, W.; Zhou, J.; Wang, Z.; Zhou, X.; Zhou, F.; Sun, J.; Song, M.; Zhou, Z. Three-Dimensional Object Detection Network Based on Multi-Layer and Multi-Modal Fusion. *Electronics* **2024**, *13*, 3512. <https://doi.org/10.3390/electronics13173512>

Received: 25 June 2024

Revised: 26 August 2024

Accepted: 27 August 2024

Published: 4 September 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: auto-driving; 3D object detection; multi-sensor fusion; self-attention mechanism

1. Introduction

With the rapid development of artificial intelligence, research on intelligent unmanned systems is no longer limited to performing simple mechanical tasks, and further focus is on the interaction of intelligent unmanned systems and an unpredictable world. From a biological perspective, the sensory organs and nervous system of animals receive multimodal physiological signals and send them to the central nervous system for processing, using the brain to make behavioral decisions. The sensorimotor system of animals is complex and consists of many small structures. Each small structure has only a single function, and when the number of structures forms a certain scale, the interaction effects between them increase dramatically. Through these large-scale and complex interactions, biological systems perceive and make decisions based on the external environment, showing intelligent abilities called emergent abilities.

Emergent abilities can provide inspiration and reference for unmanned systems, which are self-adaptive and innovative, and can adjust and learn according to environmental changes. In recent years, with the improvement of hardware and the rise of artificial intelligence technology, driving is being increasingly intelligent and unmanned. Automatic driving systems consist of three parts: perception, decision, and control. The perception

system refers to the sensors carried by the automatic driving system, including vision sensors, LiDAR, and millimeter-wave radar. It provides accurate information about objects and is the basis for autonomous navigation and obstacle avoidance. By integrating information from multiple sensors, multi-modal fusion algorithms can provide a more comprehensive understanding of the environment and enable unmanned systems to make more informed decisions, effectively enhancing the perception and interaction capabilities of unmanned systems. Among them, multi-modal object detection is an important part of unmanned system perception, which provides basic information for the unmanned system to make decisions.

Object detection methods mainly include 2D detection, 3D LiDAR-only detection, 3D camera-only detection, camera–LiDAR fusion detection, etc. From the perspective of sensor characteristics, vision sensors can obtain rich semantic feature information of objects, but the imaging quality is easily affected by background interference, and the depth information of the detected object cannot be obtained when it is occluded. In contrast, LiDAR can obtain the position and speed of the object and is almost unaffected by environment illumination, but it cannot provide color, texture, or other visual information. Moreover, with increases in the target distance, the sparsity of the point cloud become more obvious. The strong complementarity between LiDAR and vision sensors makes it achieve information complementarity and higher-precision object detection through multi-sensor fusion. However, the current research shows that in the detection capability ranking of the KITTI dataset, LiDAR-only methods such as PointPillars [1], PointRCNN [2], and VoxelNet [3] generally perform better than camera–LiDAR fusion methods such as Frustum PointNets [4], MV3D [5], and AVOD [6].

According to the stage of fusion, multi-sensor fusion methods can be roughly divided into three categories. Early fusion methods usually process raw data directly, which can be regarded as data-level fusion. Such methods require precise alignment of data from different sensors, and if the raw data do not match well, the resulting misalignment will lead to a severe degradation in the detection performance. Typical early fusion methods include PointPainting [7] and PI-RCNN [8]. Although this method realizes the transfer of image semantics to point cloud space, it also transfers noise from one mode to another. This noise, combined with the alignment of features in the point cloud, obviously destroys the saliency of some point cloud spatial features. Late fusion methods only fuse the processed features at the decision level. Since the spatial and pattern differences between the point cloud and image are significantly reduced at this stage, it is also classified as decision-level fusion. However, this fusion method is not satisfactory because the fused ROI region still contains a large amount of background noise, and how to reasonably use the multi-sensor features to generate confidence remains to be studied. Deep fusion methods are usually flexible and complex with different network structures: there are two-level cascaded fusion such as Frustum PointNets and parallel feature fusion such as MV3D and AVOD. These methods all use the original, high-level semantic information, but most of them require significant calculations, and the detection indicators are not very good.

The main reasons for the poor performance of the multi-modal fusion methods include the dislocation of the image and the point cloud perspective and the feature mismatch caused by the difference in heterogeneous data formats. Therefore, we propose a 3D object detection network based on the multi-layer and multi-modal fusion method (3DMMF), which fully integrates image features and point cloud features in multiple stages. At each fusion stage, the feature confidence of different modes is emphasized to enhance the detection effects of the fusion method. The experimental verification of this method on the KITTI dataset shows that the performance of this multi-modal and multi-layer fusion method is good. Compared with the baseline of LiDAR-only network, the performance is obviously improved.

The proposed fusion method delivers the following contributions:

- We design and implement a 3D object detection network based on the multi-layer and multi-modal fusion method, which paints and encodes the point cloud (Frustum RGB

PointPainting) in the frustum proposed by the 2D object detection network, therefore increasing the amount of input information in the 3D detection network.

- To solve the problem that doubling the number of channels may affect the spatial shape characteristics of the point cloud, a context-aware self-attention mechanism module is introduced to the 3D object detector, which can perceive the global image while extracting spatial features.
- CLOCs is introduced to fuse the 2D and 3D detection results without NMS to further improve the detection accuracy. Experiments on KITTI datasets prove that this fusion method has a significant improvement over the baseline of the LiDAR-only method, with an average mAP improvement of 6.3%.

2. Materials and Methods

2.1. 3D Object Detection Using Objectness

Alexe et al.'s "Measuring the Objectness of Image Windows" [9] proposes a generic objectness measurement for object detection, quantifying the likelihood of image windows containing objects of any category to distinguish between objects with clear boundaries and background elements. This method, integrating various image cues within a Bayesian framework, effectively identifies object characteristics such as closed boundaries, providing an efficient attention mechanism for a variety of image processing applications. DeepBox [10] employs CNNs to re-rank proposals generated using bottom-up methods. This approach treats "objectness" as a high-level concept, differing from traditional methods that solely rely on bottom-up cues. DeepBox adopts an innovative four-layer CNN architecture that can match larger networks in the task of evaluating objectness while being faster. RON [11] is an efficient object detection framework that combines the strengths of region-based and anchor-free methods. It achieves multi-scale object localization through reverse connections and significantly reduces the search space for negative samples by utilizing objectness priors. Through the optimization of a multi-task loss function, RON can directly predict the final detection results from various positions in the feature maps.

2.2. Three-Dimensional Object Detection Using Point Clouds

According to the different ways to process the raw point cloud, LiDAR-only networks have gradually been divided into point-based and voxel-based methods. Point-based methods directly use the raw point cloud as the feature carrier, and voxel-based methods organize the raw point cloud in a certain region with some rules and then send it into Convolutional Neural Networks (CNNs) for feature extraction after unified processing.

For point-based methods, PointNet [12] uses the arrangement invariance of point cloud spatial features, interaction and transformation invariance between point sets, and uses Multi-Layer Perceptron (MLP) to directly process a single original point cloud to maximize the retention of spatial features. PointNet++ [13] is proposed to solve the problem that PointNet does not capture the local structure induced by the metric space points and has limited generalization capabilities for complex immersions. PointRCNN [2] applies a two-stage architecture, firstly generating 3D proposals and then refining these proposals to obtain accurate detection results. Complex-YOLO [14] is proposed based on YOLOv2, and a new Euclidean region proposal network is designed. Inspired by this, YOLO3D [15] uses dual-input channels and optimizes the loss function. Point-GNN [16] introduces the idea of graph convolution for 3D object detection to adapt to the irregularity of the original point cloud. 3DSSD [17] uses a sampling strategy that combines Euclidean spatial distance and semantic features because of the few sampling points for foreground points.

For voxel-based methods, BirdNet [18] projects the 3D point cloud into a 2D bird's-eye view and uses the Faster-RCNN network to predict the detection box of the bird's-eye view. BirdNet+ [19] extracts the height and pitch angle information of the target box in advance to avoid blurring it during subsequent processing to solve the problem that BirdNet roughly processes the height information of the target. VoxelNet [3] encodes the sparse raw point cloud into dense voxel information, aggregates local features and

point-by-point features, and then extracts features using 3D CNNs. SECOND [20] uses sparse 3D CNNs for the sparsity of point cloud to improve the poor speed of VoxelNet using 3D CNNs. PointPillars [1] uses the “Pillar” method to generate pseudo-images and then uses 2D CNNs to extract features from the pseudo-images, achieving detection efficiency 2~4 times faster than previous methods. PV-RCNN [21] assumes that voxel-based operations can retain more accurate position information because of variable sensory fields, so it integrates the efficient learning of 3D voxel CNNs and the flexible receptive range of PointNet.

2.3. Three-Dimensional Object Detection Using Multi-Modal Fusion

MV3D [5] generates 3D proposals from a bird’s-eye view (BEV) map and projects them to the bird’s view, front view, and image. Then, it combines region-wise features obtained via ROI pooling for each view. Compared to MV3D, AVOD [6] removes the input of the front view and the reflection intensity of the bird’s eye view, while introducing a Feature Pyramid Network (FPN) to improve the detection accuracy for small objects. Frustum PointNets [4], Pointfusion [22], and Frustum ConvNet [23] are the representatives of 2D-driven 3D detectors. Take Frustum PointNets as an example. It firstly generates 2D object region proposals in the RGB image using a 2D CNN, then it uses these proposals to generate corresponding 3D frustums. Finally, the 3D bounding box for the object from the points in frustum is regressed. PI-RCNN [8] uses 3D object detection for the point cloud branch, uses semantic segmentation for the image branch, and then attaches the features of image semantic segmentation to the point cloud. PointPainting [7] fuses the semantic information in the image with the point cloud on the pixel level through sequential encoding. 3D-CVF [24] solves the problem of misalignment of the perspective of the image and point cloud after the bird’s-eye view projection, and it maps the image pixels to the bird’s-eye view. Frustum PointPillars [25] combines Frustum PointNets, which limits the point cloud computing space with PointPainting, expanding the mask channel. CLOCs [26] uses deep learning methods to fuse 2D and 3D object detection results on the decision level. CL3D [27] integrates with a point enhancement module, which is designed to enhance the raw LiDAR with the pseudo point, point-guided fusion module. This module is developed to find image–point correspondence at different resolutions and incorporate semantic and geometric features in a point-wise manner, including an IoU-aware head for cross-modal 3D object detection. Samal [28] presents a task-driven approach to input fusion that minimizes the utilization of resource-heavy sensors and demonstrates its application to visual–lidar fusion for object tracking and path planning.

2.4. Problem Statement

Considering the problem of misalignment between the perspective of the camera and LiDAR, we propose a simple, innovative, and effective sequential fusion method based on PointPainting: projecting each lidar point into the output of the image semantics segmentation network, then concatenating the segmentation information with the original features of the point cloud at the corresponding position. The painted points can be used in any LiDAR-only detector. Compared with previous fusion methods, PointPainting impose no constraints on 3D detection architectures, and it does not need to compute pseudo point cloud.

However, PointPainting also exposes some problems in practical applications. First, the 2D image semantic segmentation is relatively difficult. The network is complex, and it leads to long waiting times for pre-transmission. PointPainting uses the image segmentation results of adjacent time frames to paint the point cloud of this frame to overcome this problem, which harms the real-time performance of the detector. Second, the input of the 3D detection network needs to increase the number of channels to be consistent with the number of categories performing semantic segmentation. Four channels need to be expanded in the KITTI dataset, and eleven channels need to be expanded in the nuScenes [29] dataset with more categories, which significantly increases the computation

costs. Third, the noise after semantic segmentation will be introduced into the point cloud, which may easily cause false alarms. Nevertheless, this method effectively improves the 3D detection accuracy by increasing the information of each point through additional channel encoding of the point cloud.

3. Three-Dimensional Object Detection Network Based on Multi-Modal and Multi-Layer Fusion

As Figure 1 shows, 3DMMF is a multi-modal and multi-layer architecture, which takes the image and point cloud as input and uses three layers to process them. The architecture of the multi-layer fusion method is as follows:

(1) Early fusion: A 2D candidate box is proposed by the 2D image object detection network. Then, in the frustum formed by the 2D candidate box after Non-Maximum Suppression (NMS), the spatial point cloud is projected to the image, and the point cloud is painted. Sequentially, the point cloud is encoded and fused in the local area, and a recommendation channel and three color channels are added after the reflectivity channel of the point cloud.

(2) Detection backbone network: The encoded point cloud is fed to PointPillars, then the CNN is used to extract the spatial features. Meanwhile, the global self-attention [30] mechanism is introduced to extract contextual features. Two features are concatenated in series and sent to the SSD detection head to propose the 3D object detection candidates box.

(3) Late fusion: The 2D candidate box generated in step (1) and the 3D candidate box generated in step (2) are encoded into two sets of sparse tensors before non-maximum suppression (NMS) and input into the camera–LiDAR object candidates fusion network (CLOCs) to obtain the final detection results.

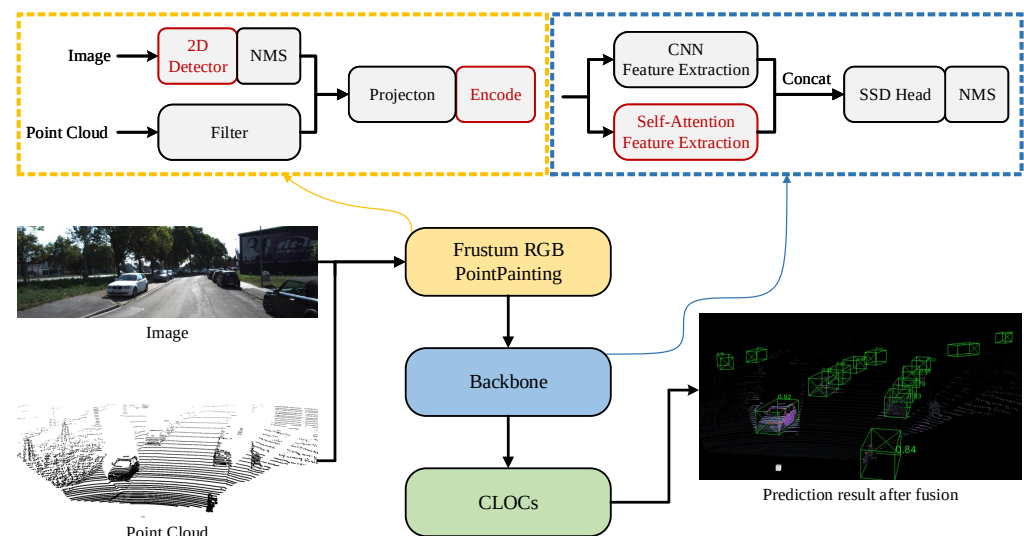


Figure 1. The architecture of the proposed multi-layer fusion method.

3.1. Early Fusion: Frustum RGB PointPainting (FRP)

On the basis of PointPainting, Frustum RGB PointPainting (FRP) is a sequential and serial structure, as shown in Figure 2.

We replace the image semantics segmentation network in PointPainting with a 2D object detection network. Firstly, the semantics segmentation task needs to identify and classify the image pixels one-by-one and output the pixel-level masks of different object categories, while the pixel-level image labeling task is more cumbersome than labeling the detection frame. Secondly, a simple segmentation task requires only one bool value to indicate whether an object exists, but it becomes complicated when it comes to images containing multiple categories. There are multi-category objects in the KITTI dataset without time labels. If the image segmentation processing time is too long, the pre-transmission

time cannot be reduced. When the vehicle is driving too fast, the 3D detection network still uses the image segmentation results of the previous frames to paint the encoded point cloud data, resulting in false or missed detection.

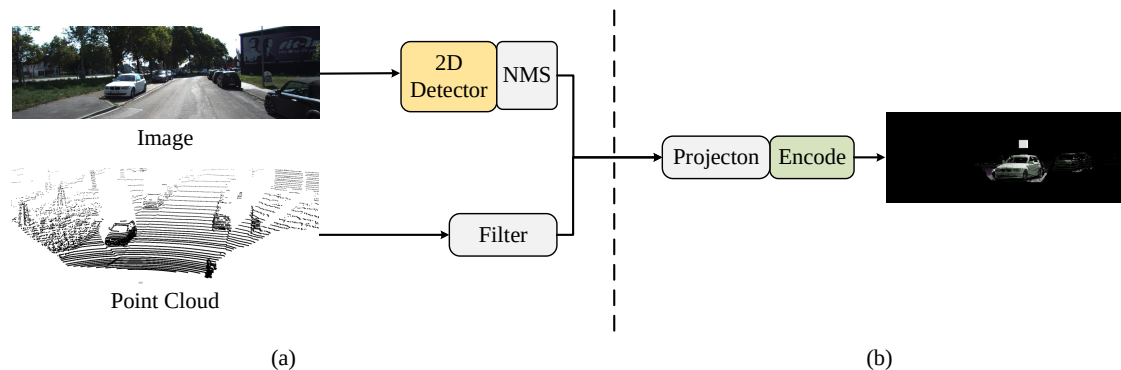


Figure 2. (a) Two-dimensional object detection using a relatively mature 2D detector and generating frustums after passing through NMS. (b) Regional data-level fusion projecting the spatial point cloud to the image, painting the point cloud in the frustum, sequentially encoding the point cloud in the local area, and adding a recommended channel and three color channels. The recommended channel is a two-dimensional Gaussian probability function mask, and the color channel is RGB information corresponding to the image pixel.

Another reason for replacing the forward serial network with a 2D object detection network is that the improvement is not good when projecting the image segmentation result into the point cloud space, especially for some objects that occupy a large number of point clouds. Frustum PointNets was used for experiments using 2D image segmentation masks to predict point cloud candidate regions. The results showed that the effects were far worse than the detection results after 3D segmentation. Even when using the common 2D and 3D segmentation results for predictions, the cumulative error caused by the 2D segmentation results will also reduce the accuracy of the prediction after 3D segmentation. Therefore, Frustum PointNets [4] assumes that in the typical 2D mask prediction task, although the estimated 2D mask is of good quality on the RGB image, it still contains significant clutter and foreground points, which will be more obvious when projected onto 3D space. As shown in Figure 3, pixel-level segmentation errors may cause distance errors of tens of meters in the 3D point cloud space. As a result, Frustum PointNets uses a 2D object detector with a relatively good performance to generate 2D object regions and form frustums for 3D segmentation prediction instead of using projection prediction after segmentation. This method has achieved good results, and many Frustum algorithms that add restrictions to point cloud information have been developed on the basis of this work. Among them, the newly published Frustum PointPillars method combines the idea of limiting the calculation space for point cloud and adding the mask channel to point clouds, proposing an innovative method of adding a “probability mask” in the frustum. This is the basis of the idea for adding the “recommended channel” to the point cloud in this paper.

The 2D detector in this paper is YOLOv5, which is currently one of the preferred detection algorithms in the engineering community, with the advantages of a clear architecture and good real-time performance. Compared with the two-stage detection network such as Fast R-CNN and the single-stage detection network such as SSD, etc., YOLOv5 improves the detection speed while maintaining the accuracy. The ultra-fast processing speed is more in line with the early fusion serial processing structure of FRP, which reduces the time consumed in the 2D image-processing stage, thereby reducing the waiting time for pre-transmission. YOLOv5 uses CSPDarknet53 as its backbone network and FPN to fuse information from different feature map levels, outputting feature layers of three scales: 19×19 , 38×38 , and 76×76 , taking into account the detection robustness of multi-scale objects. In the experiment, the 2D annotation file format of the KITTI dataset is first converted

to the VOC dataset format, because YOLOv5 uses the VOC dataset format for training. Then, YOLOv5 is trained, and finally, the data prepared for fusion are obtained.

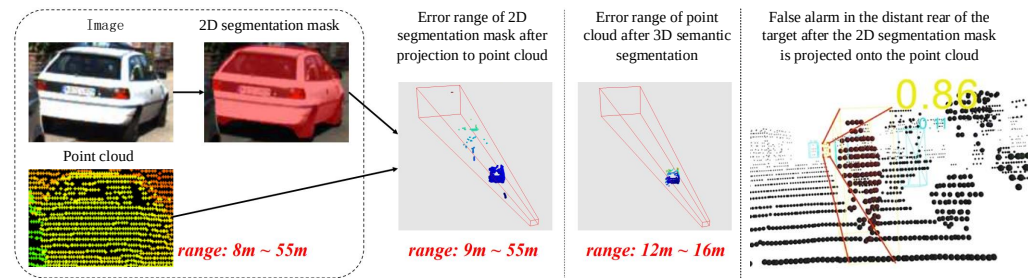


Figure 3. False detection caused by 2D mask noise in the rear of 3D space.

This fusion method adds four encoding channels to each point cloud and expands the raw KITTI point cloud information from (x, y, z, r) to (x, y, z, r, S, R, G, B) , where x, y , and z are the spatial position information of the point cloud, r is the laser radar reflection intensity, S is the recommended channel, and R, G , and B are the color channels corresponding to the point cloud.

The preprocessing of the point cloud mainly consists of two parts: point cloud filtering and data enhancement. Using the preprocessed point cloud for training can reduce the computation cost and improve the generalization performance of the algorithm. Point cloud projection needs to associate the point cloud coordinate system with the image coordinate system. This process needs to complete the conversion from the LiDAR coordinate system to the pixel coordinate system.

Data augmentation improves the generalization performance and detection effects of the detector by increasing the number of samples. Considering that there are no objects in the point cloud of most regions in the KITTI dataset, in order to ensure balanced training samples, we firstly generate point cloud for all types of objects based on the reference bounding box lookup table and box. According to the annotation information of the 3D bounding box, a sample library is constructed using all the sample data. The sample library stores the annotation information of the point cloud and bounding boxes of various types of vehicles, pedestrians, and cyclists. Real samples of 15 cars and 8 cyclists are selected to join the current point cloud data for subsequent training. Secondly, the reference boxes of the real target are added one-by-one, and the frequency shift operation or rotation operation is performed on each box, wherein the translation operation is $N(0, 0.2)$ times of the normal distribution of the x, y, z coordinate, and the rotation operation range is $[-\pi/20, \pi/20]$. After increasing the amount of training samples for the above method, the global bounding box is mirrored and flipped along the x -axis, and the translated global point cloud is drawn according to the same normal distribution $N(0, 0.2)$ to simulate positioning noise.

After filtering and enhancing the point cloud, the point cloud is projected onto the 2D image plane for painting and encoding. The process of projecting, painting, and encoding the point cloud is shown in Algorithm 1. Recommendation channel S adopts the probability mask in Frustum PointPillars, which mainly uses the object distribution probability information. We know that the center of the 2D detection box is more likely to be occupied by objects, and the projected 3D points in the central area are more likely to belong to objects, not background clutter. This method defines the probability of points belonging to an object as a Gaussian function:

$$L(\bar{x}, \bar{y}) = \exp \left(-\frac{(\bar{x} - \bar{x}_0)^2}{2w^2} - \frac{(\bar{y} - \bar{y}_0)^2}{2h^2} \right) \quad (1)$$

After the recommended likelihood value is added to the reflection intensity r of the original point cloud as an additional feature vector, the maximum likelihood value is selected if one point is shared by multiple 2D boundary boxes at the same time.

Algorithm 1 Frustum RGB PointPainting**Input:**

LiDAR point cloud $L \in \mathbb{R}^{N,D}$ (N is the number of points, D is the dimension, typically 4 for KITTI dataset.)

Recommended channel $S \in \mathbb{R}^{W,H}$

Color channel $C \in \mathbb{R}^{W,H,3}$ for RGB channels

Extrinsic transformation matrix $M_2 \in \mathbb{R}^{4,4}$

Camera intrinsic matrix $M_1 \in \mathbb{R}^{3,4}$

Method:**for** $l \in L$ **do**

Point cloud projection:

$$\vec{l}_{image} = \text{Projection}(M_1, M_2, \vec{l}_{point(x,y,z)})$$

$$\vec{l}_{image} \in \mathbb{R}^2$$

Recommended Channel acquisition:

if Point $\in$ boxes generated by 2D detection **then**

$$S = L(\bar{x}, \bar{y})$$

else

$$S = 0$$

end if

$$\vec{s} = S(\vec{l}_{image}[0], \vec{l}_{image}[1]) \quad \vec{s} \in \mathbb{R}$$

Color channel acquisition:

$$\vec{c} = C(\vec{l}_{image}[0], \vec{l}_{image}[1]) \quad \vec{c} \in \mathbb{R}^3$$

Point cloud encode:

$$\vec{p} = \text{Encode}(\vec{l}, \vec{s}, \vec{c}) \quad \vec{p} \in \mathbb{R}^{D+1+3}$$

end for**Output:**

$$\text{Encoded point cloud } P \in \mathbb{R}^{N,D+1+3}$$

The sequential encoding method adopted by FRP has two main characteristics: local encoding of the frustum and RGB color encoding. Global point cloud encoding corresponds to the local encoding of the frustum, but the experiments prove that the detection performance of the 3D detection network after global point cloud encoding is not significantly improved compared with the original baseline, or it even declines. We find that the global color information will submerge the more important spatial coordinate information for the point cloud, leading to the degradation of the spatial information's salience. Local encoding of the frustum forms a recommended region of the object, then RGB color encoding is performed, as well as recommended channel encoding in this region. It retains the semantics of the object position and can also attach additional color, material, brightness, and shape information to the point cloud.

FRP requires the 2D detector to provide relatively accurate object position semantics, and candidate boxes should be located accurately, using as few as possible. Therefore, before the completion of fusion encoding, the detection results of YOLOv5 need to be introduced into non-maximum suppression (NMS). Additionally, a local optimal solution search should be carried out in the domain space of the candidate box, forming a frustum in the region with the highest probability that 2D objects will appear. The candidate boxes that are not introduced into NMS will be directly transmitted to CLOCs.

3.2. Three-Dimensional Object Detection Network Based on Self-Attention Mechanism

The 3D backbone network used in 3DMMF is modified based on PointPillars. Firstly, it converts the point cloud into sparse pseudo-images in the form of vertical “pillars”, then it uses 2D CNNs to detect the pseudo-image and predict the 3D detection boxes. In the early fusion stage, the channel dimension of the raw point cloud is expanded, and the input size of the 3D object detector needs to be modified accordingly. In addition, the number of point cloud channels after encoding is doubled, also doubling the amount of feature

information represented by a single position of the pseudo-image when extracting features to form sparse pseudo-images from the “pillars” tensor formed by the point cloud. This will blur the spatial shape feature of the object’s point cloud to a certain extent, resulting in some high-confidence false alarms. In order to adapt to the point cloud after FRP, the depth of the convolution layer for feature extraction must be increased so that the receptive field is large enough to capture the encoded point cloud information.

However, the relative importance of the input point cloud features deepens as the network progresses downward, and optimizing coordinate parameters across multiple layers to capture patterns in the data is challenging. Taking this into consideration, we introduce a context-aware self-attention mechanism module for PointPillars. The core idea of the self-attention mechanism is that it firstly obtains the global point cloud information, then it weights and sums the features of all point cloud positions to the object position and dynamically calculates the corresponding weights based on the similarity function between the features of each position in the embedding space. This feature extractor can learn the correlations between the global cloud points to produce more powerful, unique, and robust features. For example, there is an obvious correlation between the features of cars and bicycles driving in the same lane and the height and color features of pedestrians. The context-aware self-attention mechanism can extract the correlation of such features of road objects and take advantage of the color point cloud information to produce more accurate detection, especially for cars that are farther away, smaller pedestrians, etc.

As Figure 4 shows, the channel-expanded PointPillars are mainly divided into three stages: (1) convert the point cloud encoded by FRP into a pseudo-image sparse tensor; (2) use the spatial feature extractor based on the CNN backbone network and the contextual feature extractor based on the self-attention mechanism module to extract pseudo-image features; (3) use the detection head to predict and regress 3D detection boxes.

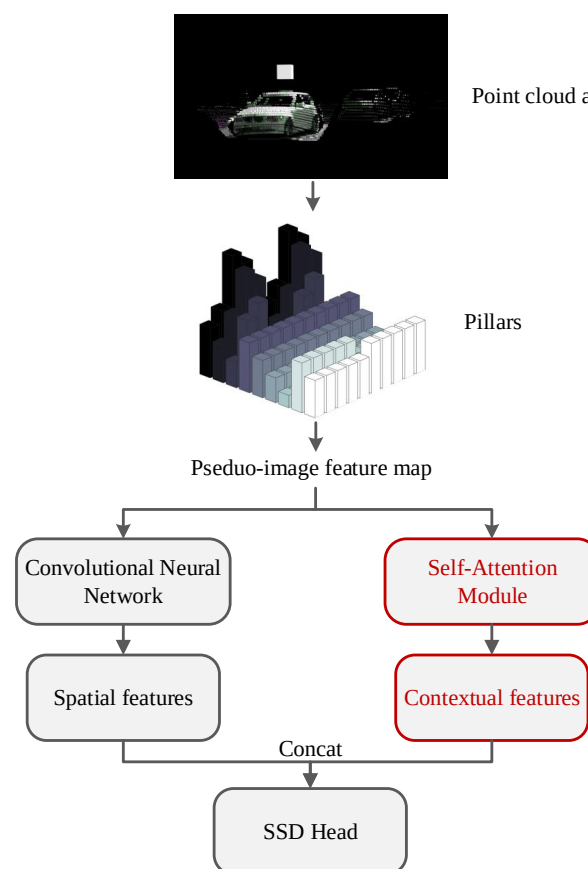


Figure 4. The improved 3D object detection network structure introducing a context-aware self-attention mechanism module.

As Figure 5 shows, PointPillars will divide the input point cloud for representation, then distribute $H \times W$ grids of size $0.16 \text{ m} \times 0.16 \text{ m}$ uniformly and continuously in the grid in the Oxy plane to form a set of “pillars”. At the same time, the 8-dimensional point cloud (x, y, z, r, S, R, G, B) encoded by FRP in the early fusion will be enhanced and represented as 13-dimensional $(x, y, z, r, x_c, y_c, z_c, x_p, y_p, S, R, G, B)$, where x, y , and z are the position coordinates of the raw point cloud, r is the reflectivity, x_c, y_c , and z_c are the arithmetic mean centers of all the points in the “Pillars”, x_p, y_p represent the offset of the arithmetic center from the center of the “pillars”, S is the recommended channel, and R, G , and B represent the color channels corresponding to the point cloud. By limiting the number P of non-empty “pillars” of each sample and the number of points N in each “pillar”, the sparsity is maintained from 0 to 97%. If the amount of data in the “pillars” is large, random sampling will be performed. On the contrary, if the amount is too small, it will be filled with “0”. Therefore, a tensor of size (D, P, N) is established, where D represents the feature dimension of each point cloud, which is 13 after FRP encoding (the original PointPillar is 9), P represents all non-empty pillars, and N represents the maximum number of point clouds in each pillar. Then, a simplified end-to-end network PointNet is used to further process and extract features from the tensorized point cloud data. It applies the data of each pillar into a linear layer for dimension-up operation and generates a tensor with the size of (C, P, N) through the BatchNorm layer and ReLU layer. Then, it obtains a tensor with the size of (C, P) after performing max pooling on the channel. Finally, the encoded features are piled up according to the position of the original pillar and scattered back to the original “pillar” position, creating a pseudo-image feature map with the size of (C, H, W) , where $C = 64$ is the number of channels, and H and W represent the height and width of the pseudo-image.

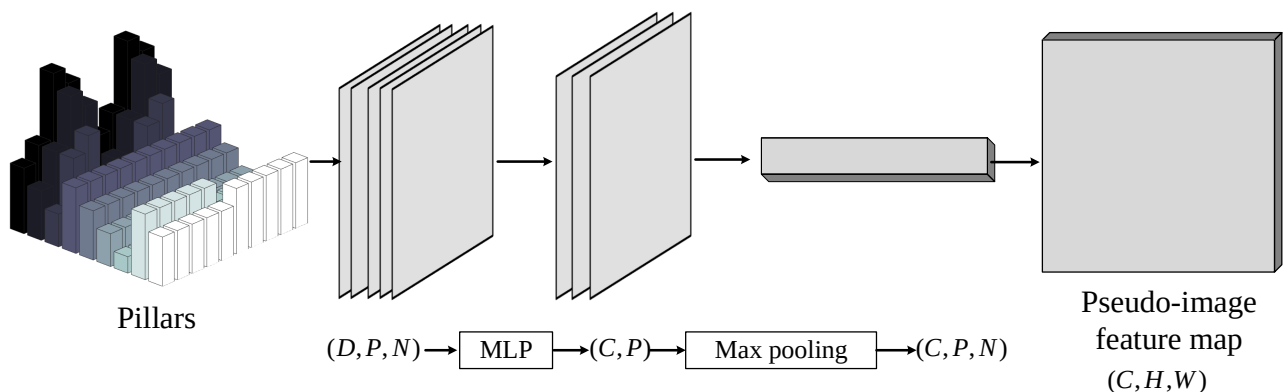


Figure 5. Pseudo-image generation process.

After introducing the self-attention mechanism, we use both the spatial feature extractor based on the CNN backbone network and the contextual feature extractor based on the self-attention mechanism to extract more comprehensive and effective global point cloud features.

A CNN backbone network similar to FPN is used for spatial feature extraction, as shown in Figure 6. The network is divided into a downsampling connection network and an upsampling connection network. In total, three layers of convolution operations are performed to complete the multi-scale feature extraction and channel splicing fusion. The downsampling network extracts more abstract and high-level pseudo-image features through convolution while reducing the spatial resolution. The upsampling network keeps the feature image size aligned with the original pseudo image through deconvolution.

Before the global self-attention mechanism module extracts the context features, it is essential to further express the pseudo-image features $(X = x_1, x_2, \dots, x_n)$ extracted by the “pillar” mathematically to form a connection graph $G = (v, \varepsilon)$. The node set in the graph $v = (x_1, x_2, \dots, x_n \in \mathbb{R}^c)$ is composed of pseudo-image features, connecting the nodes

in pairs to form an edge set $\varepsilon = r_{ij} \in \mathbb{R}^{N_h}, i = 1, \dots, n, j = 1, \dots, n$, where the edge r_{ij} represents the relationship between the i -th node and the j -th node, and the N_h space is the total number of attention maps. After obtaining the feature node set v from the attention module, the edge set ε will be calculated. This method of representing point cloud features as nodes in the connection graph will transform the process of aggregating global context information into a process similar to tracing the information transfer between nodes on the connection graph, which is more conducive to extracting high-order interactions between global cloud points.

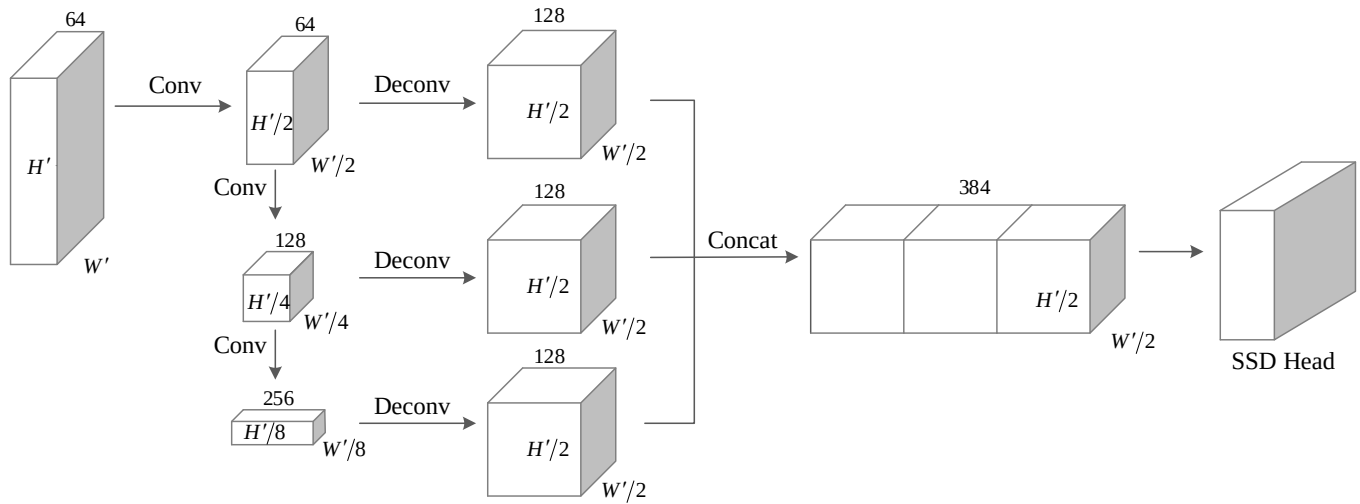


Figure 6. Spatial feature extractor based on the CNN backbone network.

As Figure 7 shows, based on the global self-attention mechanism module, the processed feature map is mapped to the query vector Q , key vector K , and value vector V through the linear layer, where V represents the input feature, and K and Q are used to compute the importance of the feature in the attention map. We calculate the similarity dot product between query element q_i and each key $k_{j=1:n}$ to obtain the weight:

$$\begin{cases} f(Q, K_i), & i = 1, 2, \dots, n \\ f(Q, K_i) = \frac{Q^T K_i}{\sqrt{D}} \end{cases} \quad (2)$$

We use the softmax function to normalize these weights and convert them to attention weights w_i :

$$w_i = \frac{e^{f(Q, K_i)}}{\sum_{j=1}^m f(Q, K_j)}, i = 1, 2, \dots, m \quad (3)$$

The pairwise interaction terms $r_{ij} = w_{ij}v_j$ in the connection graph are calculated using the attention weights. The cumulative global context information of each node vector a_i is the sum of these pairwise interaction terms. Multiple parallel attention heads can independently extract channel dependencies. Concatenating this cumulative global context vector across attention heads produces the final output $a_i^{h=1:N_h}$ of node i , passes it through a linear layer, normalizes using layer normalization, then uses x_i of the original input feature map to perform residual calculation connection $F(x_i) + x_i$ to calculate the result of the global self-attention mechanism module.

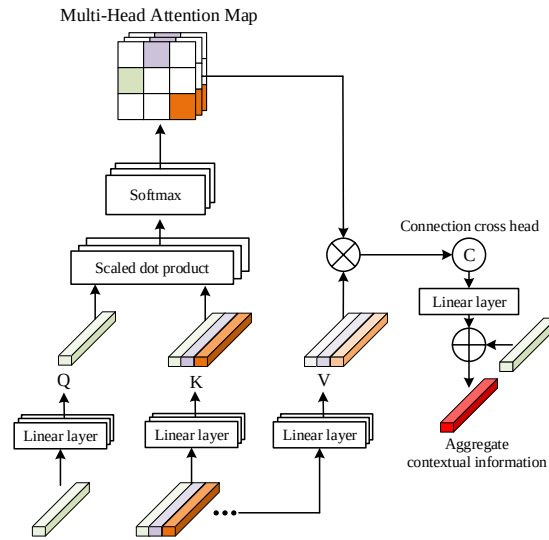


Figure 7. Context-aware self-attention mechanism module.

3.3. Late Fusion: CLOCs

The method used in the late fusion stage is Camera-LiDAR Object Candidates Fusion Networks (CLOCs), which is the first decision-level fusion method based on deep learning in a strict sense. According to the principle that the detection results of different sensors for the same object will be consistent in geometry and semantics and the selection of candidate detection results may be wrongly suppressed by a single-modal method, CLOCs use the probability drive to choose the spatial position and category semantics of 2D detection boxes and 3D detection boxes, obtaining more accurate detection results. Geometric consistency holds that if an object is detected simultaneously in both 2D and 3D, there will be at least one highly overlapping bounding box on the image plane. It is difficult for a single-modal false alarm object to have a bounding box in multiple modes at the same time, so the geometric consistency can be quantified based on the intersection-over-union (IOU) ratio between the 2D and 3D candidates projected onto the 2D plane to judge whether a false alarm occurs. Semantic consistency can be considered as the ability of a single modality to detect multiple objects, but CLOCs perform association fusion on the same type of object.

The main network structure of CLOCs is shown in Figure 8. For data in KITTI, 2D candidates and 3D candidates are defined as follows:

$$p^{2D} = \{p_1^{2D}, p_2^{2D}, \dots, p_k^{2D}\} \quad (4)$$

$$p_i^{2D} = \{[x_{i1}, y_{i1}, x_{i2}, y_{i2}], s_i^{2D}\} \quad (5)$$

$$p^{3D} = \{p_1^{3D}, p_2^{3D}, \dots, p_n^{3D}\} \quad (6)$$

$$p_i^{3D} = \{[h_i, w_i, l_i, x_i, y_i, z_i, \theta_i], s_i^{3D}\} \quad (7)$$

k 2D candidates and n 3D candidates will be encoded into a $k \times n \times 4$ tensor, and each element T_{ij} corresponds to four channels:

$$T_{ij} = \{IOU_{ij}, S_i^{2D}, s_i^{3D}, d_j\} \quad (8)$$

In the formula, IOU_{ij} is expressed as the intersection-over-union, which is calculated from the i -th 2D candidate result and the j -th 3D candidate result, which is the concentrated expression of geometric consistency. s_i^{2D} is the confidence of the corresponding box, s_i^{3D} is the confidence of the corresponding box, and d_j denotes the normalized distance from the j -th 3D candidate bounding box to the ground Oxy . This representation can easily

eliminate tensors with low *IOU* values to reduce false alarms. T is processed by four convolution layers to produce a tensor of size $1 \times p \times 1$. Then, a tensor T_{out} of size $k \times n \times 1$ is constructed by filling p output elements with reference to the position index (i, j) of the input tensor. Finally, by max-pooling the first dimension of the tensor T_{out} to map it with the expected target of the probability score of $1 \times n$, the final fusion result will be selected.

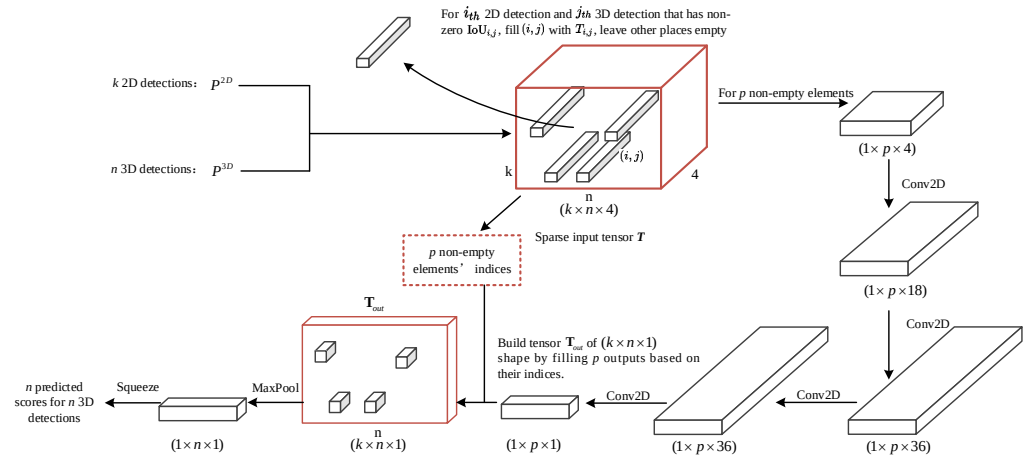


Figure 8. The main network structure of CLOCs is as follows: (1) 2D and 3D detectors propose non-NMS candidate boxes, respectively, (2) the candidates of these two modalities are encoded into sparse tensors, (3) 2D convolution is used to complete the corresponding feature fusion, and (4) the processed tensor is mapped to the desired learning target through max-pooling, which is the probability score mapping. The process of encoding the candidates of the two modalities into a sparse tensor requires generating joint detection candidates of the same object.

In this paper, the late fusion method uses the focal loss function for object classification. This loss function adds a factor $(1 - p_t)^\gamma$ to the standard cross-entropy, making the model more focused on a small number of target categories and increasing the weight of misclassified samples in order to correct the category imbalance and ameliorate the generalization performance of this network. The standard binary classification Cross Entropy (CE) loss can be expressed as follows:

$$CE(p, y) = \begin{cases} -\log p, & y = 1 \\ -\log(1 - p), & \text{others} \end{cases} \quad (9)$$

In the formula, y represents the true class: “1” for the positive class and “0” for the negative class. $p \in [0, 1]$ is the predicted probability of the model for the label $y = 1$ class. For notation convenience, we defines p_t as follows:

$$p_t = \begin{cases} p, & y = 1 \\ 1 - p, & \text{others} \end{cases} \quad (10)$$

Then, the CE can be expressed as follows:

$$CE(p_t) = -(\alpha_t) \log p_t \quad (11)$$

The focal loss is as follows:

$$FL(p_t) = -(\alpha_t)((1 - p_t)^\gamma) \log p_t \quad (12)$$

We set $(\alpha)_t = 0.25$, $\gamma = 2$ in the experiments to train.

4. Experimental Results

4.1. Experimental Environment

To verify the advantages of the 3D object detection network based on the multi-layer and multi-modal fusion method (3DMMF) compared with other methods, we will perform an experimental evaluation of the KITTI dataset. The KITTI dataset is currently one of the world's largest computer visual public evaluation datasets for autonomous driving. The full dataset consists of 389 pairs of stereoscopic images and optical flow maps, 39.2 km visual ranging sequences, and more than 200,000 images with 3D labels, sampling, and synchronization at 10 Hz. This is currently one of the most authoritative platforms for verifying the performance of computer vision techniques. The KITTI dataset is slightly different from other public datasets, because each sensor coordinate system is set based on the driver.

The detection results are generated by the KITTI evaluation indicator plugin in OpenPCDet. The backbone network used in 3DMMF is PointPillars, the size of the pillar is $0.16 \times 0.15 \times 4$, the maximum number of cloud points in the pillar is 32, and the maximum number of pillars is 16,000 in the training set and 40,000 in the validation set. The Adam optimizer with an initial learning rate of 3×10^{-3} is used to update the variables to prevent the late learning rate from becoming too large and the result not converging. Since the global self-attention mechanism will consume significant GPU memory resources, the batch size is set to 1, and the maximum training is 80 rounds.

4.2. Detection Results and Analysis

We evaluate the performance of our method, the LiDAR-only method, and the single-stage fusion method on the KITTI dataset. Table 1 shows that 3DMMF has a better performance on the car class in all three difficulty indicators compared with the single-stage fusion method MV3D, F-PointNet, AVOD, PI-RCNN, and F-ConvNet, performing best in the easy and difficult levels. As shown in Figure 9, the detection frames generated by PointPillars after encoding using the original PointPainting method and our method are compared under the 3D point cloud scene and the corresponding image, respectively. It can be seen intuitively that the prediction confidence in PointPainting is improved in 3DMMF. Compared with the LiDAR-only method, it also has certain advantages, especially compared with the baseline method PointPillars. Our method greatly improved all the indicators, and the average mAP increased by 6.3%.

Table 1. Comparison of performance on the car class of the KITTI dataset

Model	Map	Car		
		Easy	Medium	Difficult
MV3D	62.85	71.09	62.35	55.12
MV3D (LIDAR)	56.94	66.77	52.73	51.31
F-PointNet	71.26	81.20	70.39	62.19
AVOD	65.92	73.59	65.78	58.38
PI-RCNN	76.41	84.37	74.82	70.03
SECOND	74.33	83.13	73.66	66.20
VoxelNet	66.77	77.47	65.11	57.73
PointRCNN	76.25	85.29	75.08	68.38
PointPillars	73.59	80.36	73.64	66.79
3DMMF (ours)	79.89	87.45	77.48	74.74

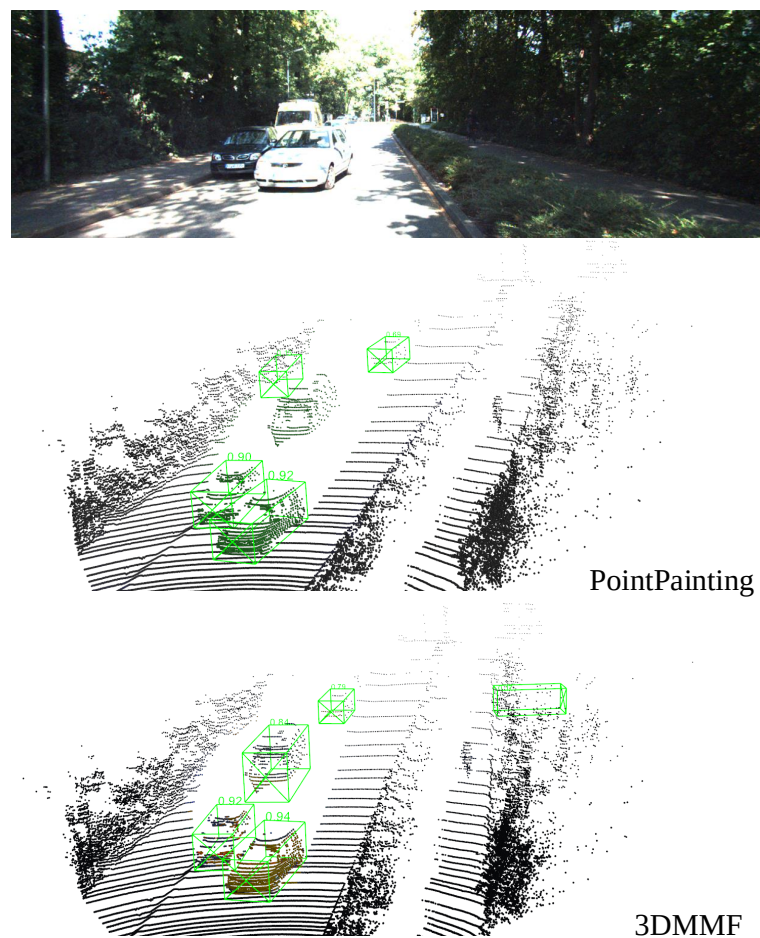


Figure 9. Prediction confidence of PointPainting and 3DMMF.

4.3. Real-Time Comparison

In terms of processing time, we divide the processing time of 3D object detection into two parts. One is the pre-transmission waiting time, which is the point cloud processing time before 3D detection. In PointPainting, it is the time for the algorithm to complete 2D image semantic segmentation and global encoding. In FRP, it is the time required for YOLOv5 to complete 2D image object detection and local encoding. The other is the 3D detection time, which is the time at which the serial 3D detection network receives the encoded point cloud to complete the 3D object detection task. A comparison of processing times is shown in Table 2.

Table 2. Comparison of processing times.

Model	Pre-Transmission Time	3D Detection Time	Total Processing Time
PointPillars	N/A	0.0202	0.0202
SECOND	N/A	0.0626	0.0626
Frustum PointNet	0.0204	0.1304	0.1508
MV3D	N/A	0.4473	0.4473
AVOD	N/A	0.1347	0.1347
PointPainting	0.4385	0.0563	0.4948
3DMMF (ours)	0.0204	0.0577	0.0781

The proposed method has the highest detection efficiency in the two-stage fusion method, while it has a good balance between detection accuracy and detection efficiency.

Compared with PointPainting, the improvement is very significant, which is brought about by transferring the image segmentation task to image detection and reducing the load of the pre-transmission task. The average pre-transmission waiting time is reduced by 95.35%. The total processing time is lower than the 3D detection time using the pipeline PointPainting method, and the real-time performance of 3DMMF can meet the needs of automatic driving.

A comparison of the effect obtained using our method and the original method is shown in Figure 10. Figure 10a,b show that, in PointPainting, a object false alarm of the car obviously appears on the left and near the LiDAR. This is because the semantic mask features of the image are very prominent, and the noise after 2D segmentation is transmitted to the point cloud. Figure 10c shows that there are false alarms, missing alarms, and objects in the wrong direction. Our method can retain the RGB color information in the frustum, and the boundary information of the target can be more accurately represented by the color drop to eliminate the problems above.

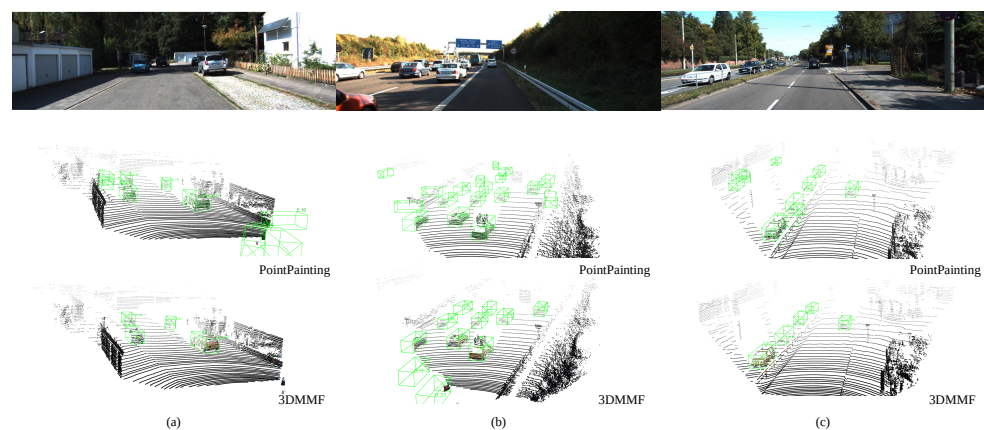


Figure 10. The effects obtained by PointPainting and 3DMMF in the figure: (a,b) In PointPainting, a object false alarm of the car obviously appears on the left and near the LiDAR. (c) In PointPainting, there are false alarms, missing alarms, and objects in the wrong direction.

4.4. Ablation Study

The proposed fusion method aims to fuse 2D and 3D information in multiple stages. Compared with the LiDAR-only baseline method PointPillars or the single-stage fusion method PointPainting, the multi-layer multi-modal fusion method can effectively improve the problem of perspective misalignment and feature mismatch. In the ablation experiments, the effectiveness of FRP local encoding in the early fusion stage, the self-attention mechanism in 3D object detection to extract contextual information, and the probability-driven decision fusion of CLOCs in the late fusion stage are demonstrated, respectively, as shown in Table 3.

Table 3. Comparison of performance in ablation experiments.

Model	FRP	FSA	CLOCs	mAP	Car		
					Easy	Medium	Difficult
PointPillars				73.59	80.36	73.64	66.79
PointPainting				73.46	79.42	73.67	67.28
	✓			74.83	80.78	74.12	69.59
		✓		77.49	85.09	75.18	72.21
Model			✓	75.62	82.26	76.12	68.49
	✓	✓		79.62	87.34	77.16	74.35
	✓	✓	✓	79.89	87.45	77.48	74.74

5. Conclusions

The application and development of unmanned systems can be further promoted by studying and solving the existing problems of perception systems. In this paper, we propose a 3D object detection network based on the multi-layer and multi-modal fusion method to ameliorate the problems of perspective dislocation and heterogeneous feature mismatch, which are difficult to solve for the single-stage fusion method and LiDAR-only method.

In the early fusion stage, the partial sequential fusion encoding method, Frustum RGB PointPainting (FRP), is used to locally paint and encode the point cloud in the frustum formed by the 2D detection frames. Then, the 3D object detection backbone network adopts PointPillars using the context-awareness module, which integrates the global self-attention mechanism to overcome the high false alarm rate caused by the multiplication of parameters after encoding point cloud and the difficulty of extracting global point cloud features. In the late fusion stage, 2D and 3D candidate boxes are encoded as two sets of sparse tensors before significant suppression occurs. The final 3D object detection result is obtained using CLOCs to improve the overall accuracy of 3D object detection by taking advantage of the geometric space and semantic consistency of the 2D and 3D object detection results. Experiments were performed on the KITTI dataset, and the results demonstrate the effectiveness of this approach. Compared with the baseline method, PointPillars, the 3D object detection accuracy is significantly improved, and the average mAP is increased by 6.3%. Compared with the original early fusion method PointPainting, the detection efficiency of 3DMMF is improved by 84.22%, allowing the detection accuracy and efficiency to reach a more appropriate balance. Compared with other single-stage fusion methods or LiDAR-only methods, our method has obvious advantages. This multi-layer and multi-modal fusion method contributes to the development of sensor fusion to a certain extent.

Author Contributions: Conceptualization, W.Z. and J.Z.; methodology, Z.Z. and J.Z.; software, Z.W.; validation, J.S.; formal analysis, J.Z.; investigation, W.Z.; resources, J.Z. and F.Z.; data curation, W.Z.; writing—original draft preparation, W.Z. and M.S.; writing—review and editing, W.Z., M.S., and X.Z.; visualization, F.Z.; supervision, X.Z. and Z.Z.; project administration, X.Z. and Z.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: Author Wenming Zhu, Zizhe Wang, Feng Zhou and Jingwen Sun were employed by the company Changzhou Power Supply Branch of State Grid Jiangsu Electric Power Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Lang, A.H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; Beijbom, O. Pointpillars: Fast encoders for object detection from point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 12697–12705.
2. Shi, S.; Wang, X.; Li, H. Pointrcnn: 3D object proposal generation and detection from point cloud. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 770–779.
3. Zhou, Y.; Tuzel, O. Voxnet: End-to-end learning for point cloud based 3D object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4490–4499.
4. Qi, C.R.; Liu, W.; Wu, C.; Su, H.; Guibas, L.J. Frustum pointnets for 3D object detection from rgb-d data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 918–927.
5. Chen, X.; Ma, H.; Wan, J.; Li, B.; Xia, T. Multi-view 3D object detection network for autonomous driving. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1907–1915.
6. Ku, J.; Mozifian, M.; Lee, J.; Harakeh, A.; Waslander, S.L. Joint 3D proposal generation and object detection from view aggregation. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 1–8.

7. Vora, S.; Lang, A.H.; Helou, B.; Beijbom, O. Pointpainting: Sequential fusion for 3D object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 4604–4612.
8. Xie, L.; Xiang, C.; Yu, Z.; Xu, G.; Yang, Z.; Cai, D.; He, X. PI-RCNN: An efficient multi-sensor 3D object detector with point-based attentive cont-conv fusion module. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 12460–12467.
9. Alexe, B.; Deselaers, T.; Ferrari, V. Measuring the Objectness of Image Windows. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 2189–2202. [[CrossRef](#)] [[PubMed](#)]
10. Kuo, W.; Hariharan, B.; Malik, J. DeepBox: Learning Objectness with Convolutional Networks. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 2479–2487. [[CrossRef](#)]
11. Kong, T.; Sun, F.; Yao, A.; Liu, H.; Lu, M.; Chen, Y. RON: Reverse Connection with Objectness Prior Networks for Object Detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 5244–5252. [[CrossRef](#)]
12. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3D classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
13. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Adv. Neural Inf. Process. Syst.* **2017**, *30*. [[CrossRef](#)]
14. Simon, M.; Milz, S.; Amende, K.; Gross, H.M. Complex-yolo: Real-time 3D object detection on point clouds. *arXiv* **2018**, arXiv:1803.06199.
15. Ali, W.; Abdelkarim, S.; Zidan, M.; Zahran, M.; El Sallab, A. Yolo3d: End-to-end real-time 3D oriented object bounding box detection from lidar point cloud. In Proceedings of the European Conference on Computer Vision (ECCV) Workshops, Munich, Germany, 8–14 September 2018; pp. 716–728.
16. Shi, W.; Rajkumar, R. Point-gnn: Graph neural network for 3D object detection in a point cloud. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 1711–1719.
17. Yang, Z.; Sun, Y.; Liu, S.; Jia, J. 3Dssd: Point-based 3D single stage object detector. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11040–11048.
18. Beltrán, J.; Guindel, C.; Moreno, F.M.; Cruzado, D.; Garcia, F.; De La Escalera, A. Birdnet: A 3D object detection framework from lidar information. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 3517–3523.
19. Barrera, A.; Guindel, C.; Beltrán, J.; García, F. Birdnet+: End-to-end 3D object detection in lidar bird’s eye view. In Proceedings of the 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), Rhodes, Greece, 20–23 September 2020; pp. 1–6.
20. Yan, Y.; Mao, Y.; Li, B. Second: Sparsely embedded convolutional detection. *Sensors* **2018**, *18*, 3337. [[CrossRef](#)] [[PubMed](#)]
21. Shi, S.; Guo, C.; Jiang, L.; Wang, Z.; Shi, J.; Wang, X.; Li, H. Pv-rcnn: Point-voxel feature set abstraction for 3D object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 10529–10538.
22. Xu, D.; Anguelov, D.; Jain, A. Pointfusion: Deep sensor fusion for 3D bounding box estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 244–253.
23. Wang, Z.; Jia, K. Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 1742–1749.
24. Yoo, J.H.; Kim, Y.; Kim, J.; Choi, J.W. 3D-cvf: Generating joint camera and lidar features using cross-view spatial feature fusion for 3D object detection. In Proceedings of the Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Proceedings, Part XXVII 16; Springer: Berlin/Heidelberg, Germany, 2020; pp. 720–736.
25. Paigwar, A.; Sierra-Gonzalez, D.; Erkent, Ö.; Laugier, C. Frustum-pointpillars: A multi-stage approach for 3D object detection using rgb camera and lidar. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 2926–2933.
26. Pang, S.; Morris, D.; Radha, H. CLOCs: Camera-LiDAR object candidates fusion for 3D object detection. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 10386–10393.
27. Lin, C.; Tian, D.; Duan, X.; Zhou, J.; Zhao, D.; Cao, D. CL3D: Camera-LiDAR 3D object detection with point feature enhancement and point-guided fusion. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 18040–18050. [[CrossRef](#)]
28. Samal, K.; Kumawat, H.; Saha, P.; Wolf, M.; Mukhopadhyay, S. Task-Driven RGB-Lidar Fusion for Object Tracking in Resource-Efficient Autonomous System. *IEEE Trans. Intell. Veh.* **2022**, *7*, 102–112. [[CrossRef](#)]

29. Caesar, H.; Bankiti, V.; Lang, A.H.; Vora, S.; Liong, V.E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; Beijbom, O. nuscenes: A multimodal dataset for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11621–11631.
30. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.