Computing and Informatics, Vol. 45, 2026, 865–901, doi: 10.31577/cai_2026_4_865

DMS3D: MULTI-MODAL 3D OBJECT DETECTION ALGORITHM BASED ON DISCRIMINANT FUSION STRATEGY AND MULTI-SCALE FEATURE FUSION

Yan Zhang

School of Mechanical Engineering
Henan Industry and Trade Vocational College
No. 4 Yousheng North Road, Henan Province
Zhengzhou, 450053, China
e-mail: zyancau@163.com

Ruichao Xiao*, Wei Wang

School of Information Engineering
Henan Industry and Trade Vocational College
No. 4 Yousheng North Road, Henan Province
Zhengzhou, 450053, China
e-mail: mrlixubing@163.com, wangwei88@hngm.edu.cn

Lin Zhao

Anyang Cigarette Factory, China Tobacco Henan Industrial Co., Ltd.
Anyang, Henan, China
e-mail: 982493403@qq.com

Xubing Li, Zixuan Zhang

College of Computer Science and Engineering
Anhui University of Science and Technology
No. 168 Taifeng Street, Anhui Province
Huainan, 232001, China
e-mail: 2021201200@aust.edu.cn, 314663947@qq.com

Yuankun Du

School of Big Data and Artificial Intelligence
Zhengzhou University of Science and Technology
No. 1 Xueyuan Road, Mashai Industrial Park, Erqi District, Henan Province
Zhengzhou, 450064, China
e-mail: dyk049@163.com

Abstract. Multi-modal 3D object detection, which integrates point clouds and images, has gained increasing attention due to its potential to enhance detection accuracy. However, challenges such as redundant information and scale inconsistencies hinder effective feature fusion. To address these issues, we propose DMS3D (Multi-modal 3D object detection algorithm based on discriminant fusion strategy and multi-scale feature fusion), a Discriminative fusion strategy and Multi-scale feature fusion-based System for 3D object detection. Our method introduces two key modules: 1) a multi-scale feature fusion module (MSFF) based on graph neural networks, which refines point cloud representations by aligning them with image semantics while preserving fine-grained details; and 2) a discriminative fusion module (DFM) with a lightweight gating mechanism, which selectively fuses multi-modal features to reduce redundancy and noise while enhancing computational efficiency. Extensive experiments on the KITTI and nuScenes datasets validate the effectiveness of DMS3D, demonstrating superior detection performance compared to state-of-the-art methods.

Keywords: 3D object detection, LiDAR, autonomous driving, graph neural network

Mathematics Subject Classification 2010: 68T45

1 INTRODUCTION

In recent years, the significance of environment sensing technology has witnessed a remarkable surge in both industrial applications, like autonomous driving, and academic research [1, 2, 3]. Within this realm, 3D object detection holds a pivotal position, as it enables precise identification and localization of objects in three-dimensional space using sensor data such as point clouds [4], depth images [5], or LiDAR [6]. In contrast to its 2D counterpart, 3D object detection

* Corresponding author

harnesses a wealth of information, entails intricate data representation and processing, and possesses the capability to handle occlusion and multiple perspectives. These distinctive attributes bestow upon 3D object detection unparalleled advantages and underscore its utmost importance in the domain of autonomous driving.

In the initial stage, the majority of research on 3D object detection methods [7, 8, 9, 10, 11, 12] primarily focused on utilizing a single sensor (such as LiDAR or camera) or single-mode data (point cloud or image). When compared to LiDAR, cameras hold a prominent position in the realm of purely visual 3D object detection due to their cost-effectiveness and well-established image-based 2D object detection techniques. Remarkable advancements have been achieved in this field through notable endeavors such as Mono3D [5], MonoRCNN [7], and CaDDN [8], yielding commendable research outcomes. However, monocular 3D object detection poses fundamental challenges; The core problem lies in inaccurate depth estimation, making it difficult to obtain truly accurate depth information. Consequently, this limitation also results in camera-based approaches exhibiting inferior performance compared to LiDAR-based approaches. LiDAR, the most common choice in various fields such as autonomous driving or industrial robotics, can compensate for the imperfections of the camera itself. Therefore, some research methods based on LiDAR have achieved obvious performance advantages, such as VoxelNet [4], SECOND [6] and CenterPoint [9]. Although LiDAR provides accurate coordinates for environment sensing, unstructured and sparse point clouds pose certain difficulties for 3D object detection. PointNet [10] and PointNet++ [11] are the first to solve such problems. Afterwards, LiDAR-based methods perform greatly better on advanced datasets [12, 13, 14] than those using only cameras. In practice, however, LiDAR is subject to factors such as the weather. In addition, the point cloud data obtained by LiDAR lacks information such as texture and color sharpness of the image, which makes two different objects with very similar appearance structure belong to the same class when detected. In summary, both LiDAR and the camera have some drawbacks, but also complement each other. Therefore, many subsequent studies have attempted to combine the two sensors to achieve more accurate 3D object detection.

Data fusion strategies are central to multi-modal 3D object detection. Existing fusion strategies are mostly fused from feature level [15], followed by sensor level [16] and task level [17]. While these strategies improve the quality of feature fusion to some extent, there are still two major challenges. First, redundant information is introduced. Point cloud and image data commonly contain similar or repetitive information, such as different representations of the same object. When fused, if not processed, it may result in the same information appearing multiple times in the fused data, increasing the redundancy of the data. Second, point cloud and image data may have different resolutions and object size sensing ranges. In images and point clouds, the size of the same object may be represented in different ways. For example, in an image, an object may be represented in pixels, while in a point cloud, the size of an object may be represented by the density

or relative position of the points. Such size differences may make it difficult for the model to correctly understand the actual size of the object at the time of fusion.

To solve these problems, various state-of-the-art methods [18, 19, 20] use attention mechanism to fuse key information with high weight and redundant information with low weight, which effectively improves the fusion efficiency. However, the attention mechanism may increase the computational overhead of the model, especially when dealing with large-scale data, where more computational resources are required. Moreover, if there is noisy or inaccurate information in the input data, the attention mechanism may pay excessive attention to such information, resulting in model performance degradation. In addition, [12, 17] projects 3D data into 2D (such as BEV, front view, etc.), unifies the perspective, and eliminates the perspective differences that may exist between different sensors to a certain extent. In the 2D plane, the representation of point clouds is more compact, which can alleviate the data fusion problem caused by different scales. However, for complex 3D scenes, the projection method may cause severe or inaccurate information loss, resulting in degraded detection performance.

To overcome the challenges of redundant information and scale inconsistency in multi-modal 3D object detection, we propose a novel multi-scale feature fusion module (MSFF). Unlike traditional projection-based methods [13], which often lead to information loss or distortions in complex 3D scenes, our approach leverages graph-based representations of point clouds, allowing us to capture rich geometric details. By employing graph convolutions at multiple scales, we effectively preserve both local and global features of the point cloud, while aligning it with the image data on a point-by-point basis. This alignment enables the semantic features of the image to enhance the point cloud features, helping to mitigate the issues caused by scale discrepancies between modalities. Furthermore, to address the challenge of redundancy in fused data, we introduce a discriminative fusion strategy through a lightweight gating mechanism. This mechanism selectively prioritizes relevant multi-modal features and reduces the impact of redundant or noisy information, leading to more efficient and accurate detection. The combination of these innovations makes our method robust to the inherent challenges of multi-modal fusion, significantly improving detection performance.

DMS3D network mainly consists of two modules: multi-scale feature fusion module (MSFF) and discriminative fusion module (DFM). The MSFF module is a multi-scale graph convolutional neural network that fuses image information to enhance point cloud features, and it outputs scale-invariant point cloud features. This approach reduces the difference between point cloud and image scales, as the multi-scale mechanism can focus on both local details and overall features without losing key information due to scale differences. The DFM is a lightweight gating mechanism that uses multiple fully connected layers and feature concatenation operations to obtain a final fused feature for detection. This approach explicitly controls the information flow in the network structure, which helps the model to select and

fuse information from specific modalities more specifically and reduce redundancy. To assess the effectiveness of our method, we conduct evaluations on the KITTI and nuScenes datasets, specifically focusing on the significance of multi-scale feature fusion and discriminative fusion. The experimental results demonstrate the superiority of our approach in terms of performance.

Our main contributions are as follows:

- A multi-scale feature fusion module based on graph neural network is designed, which effectively alleviates the influence of point cloud scale change and rotation transformation, and makes the model obtain rich detailed features to improve the recognition of the object.
- A discriminative fusion module based on gating mechanism is proposed, which selectively fuses the features of different modes to reduce computing time, save computing resources, and reduce redundant information and noise interference.
- A large number of experiments on the KITTI and nuScenes datasets prove the superiority and excellent detection performance of the proposed DMS3D method.

The subsequent sections of this paper are structured as follows: In Section 2, a comprehensive review of the related work is presented. In Section 3, the proposed DMS3D method is introduced. Section 4 provides an in-depth analysis of the experiments conducted, while Section 5 presents the conclusions drawn from the study.

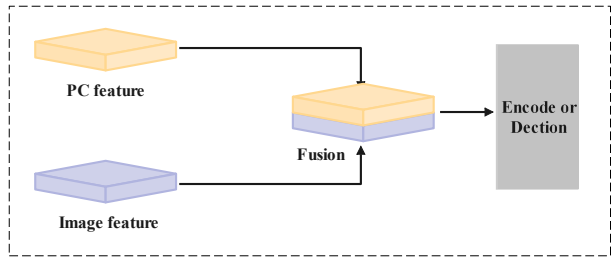
2 RELATED WORK

2.1 Single-Modal 3D Object Detection

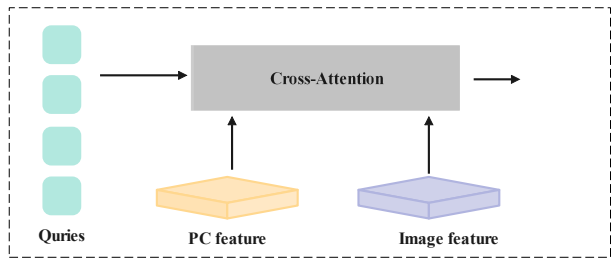
Single-modal 3D object detection holds significant prominence within the realm of 3D object detection. By relying on a single sensor or feature, this approach streamlines the data acquisition process and circumvents the intricacies associated with employing multiple sensor combinations. Over the past few years, single-modal 3D object detection has been categorized into two distinct methodologies: those based on 2D images and those based on 3D point clouds.

2.1.1 2D Images Based Method

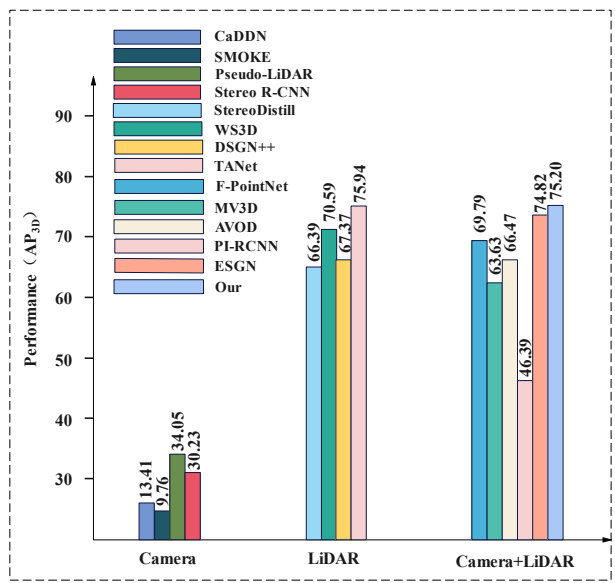
The method of 3D object detection using camera-captured images, namely estimating the 3D position, pose and size information of the object from a single image, is called monocular 3D object detection. OFT [21] uses voxel features and vertical projection to detect objects at the BEV level, but it relies on accurate depth estimation which can be affected by occlusion and different scenes. On the other hand, SMOKE [19] predicts each detected target 3D bounding box by combining a single keypoint estimation with a regression 3D variable, but insufficient feature fusion



a)



b)



c)

Figure 1. In the figure a) represents the method of directly fusing features; b) Using cross attention to fuse point cloud and image features; c) Bar charts represent camera 3D object detection, LiDAR 3D object detection, and the combination of camera and LiDAR 3D object detection methods.

may occur due to differences between different modal data. CaDDN [8] combines various feature representation methods, converts image plane into BEV, and trains depth prediction and 3D detection to improve accuracy, but it requires a complex feature alignment network, increasing computational complexity and training difficulty. Pseudo-LiDAR [22] utilizes depth information extracted from monocular or stereo images to generate a synthetic point cloud, serving as a substitute for the original LiDAR point cloud, but may introduce inaccuracies impacting detection accuracy and robustness. Stereo R-CNN [23] leverages sparse and dense, semantic, and geometric information from stereo images for 3D object detection, predicting sparse keypoints, viewpoints, and target dimensions in the stereo vision domain, and combining them with 2D bounding boxes to estimate rough 3D object boundaries. The overall image-based monocular 3D object detection method achieves the localization and detection of three-dimensional objects solely through a single image, offering promising applications in various domains. This approach eliminates the need for costly sensor devices like LiDAR, thereby reducing system expenses and enhancing practical applicability. However, due to the inherent limitations of depth information acquisition, inaccuracies may arise, along with challenges posed by occlusion and changes in perspective within the image. Consequently, object detection and localization exhibit a degree of instability. Moreover, the variations in object scale and perspective within monocular images further compound the difficulty of object detection.

2.1.2 3D Point Cloud Based Method

3D object detection method based on point cloud involves utilizing point cloud data obtained from laser radar or depth sensors for object detection. In this approach, the point cloud is treated as a collection of points within a three-dimensional space, with each point representing a sample point on the object's surface. For instance, PointNet [10] conducts feature extraction and global pooling on each point in the point cloud for overall representation, but its limitations in capturing local structures and arrangement information restrict its expressive capacity for complex point cloud data during target classification and localization. PointRCNN [24] utilizes PointNet++ for foreground-background segmentation, generating redundant bounding boxes, extracting features through operations like point cloud pooling, and refining them with initial features, but faces challenges of high computational complexity and significant memory consumption. IA-SSD [25] leverages a learnable instance-aware downsampling strategy for hierarchical foreground point selection and introduces a context-aware center perception module for accurate instance center estimation, albeit with limited capability in detecting small objects. VoxelNet [4] divides the 3D point cloud into a certain number of voxels, performs random sampling and normalization on the points for local feature extraction, utilizes 3D convolution for further feature abstraction, and finally employs the Region Proposal Network (RPN) [26] for object classification, detection, and position regression. PointPillar [27] proposes a novel point cloud encoding method for PointNet feature extraction and

subsequently maps the extracted features into a 2D pseudo-image, facilitating 3D object detection using a 2D approach. SeSame [28] enhances object detection results by utilizing semantic segmentation, which reduces information loss but may lead to limited adaptability in other scenarios due to its reliance on semantic segmentation outcomes. In summary, the 3D point cloud-based method effectively captures the precise position and shape of objects. However, the presence of noise and incomplete point cloud data can adversely impact the accuracy of object detection. Moreover, the large volume of point cloud data and the associated high processing complexity pose challenges in terms of algorithm efficiency. Finding solutions to enhance the algorithm's efficiency is an important problem that needs to be addressed.

2.2 Multi-Modal 3D Object Detection

Multi-modal 3D object detection entails leveraging diverse data types, such as LiDAR and camera, acquired from various sensors, to collectively accomplish the task of detecting 3D objects. This approach capitalizes on the strengths of different sensors, enabling the utilization of more comprehensive and abundant information, ultimately enhancing the accuracy and robustness of object detection. Presently, research in this field can be categorized into three distinct fusion methods: data-level fusion, feature-level fusion, and object-level fusion. These categories delineate different approaches for integrating and combining the data from multiple sensors at various stages of the detection process.

2.2.1 Data-Level Fusion

Data-level fusion involves the integration of raw data from diverse sensors to create a more extensive and holistic representation of the data. By combining or stitching together data at this level, data-level fusion addresses the heterogeneity between different sensor data and enhances the information available from each sensor. This comprehensive approach provides a more complete set of data features, reducing information loss and enhancing the overall performance of object detection. However, it is important to note that data-level fusion may also introduce increased data dimensionality and complexity, resulting in computational and storage overhead. Consequently, in practical applications, it becomes necessary to carefully evaluate the impact and cost of fusion, selecting appropriate fusion strategies and methods that strike a balance between the benefits gained and the associated computational and storage requirements. For example, F-PointNet [29] constructed a 3d point cloud Frustum of candidate boxes extracted from RGB images, and selected points based on the points in the Frustum. At the same time, the Pointnet++ network was used for semantic segmentation to select candidate points, and then the subsequent real box prediction was performed. F-ConvNet [30] proposed a new grouping mechanism-sliding cones to aggregate local point vector features. PointPainting [31] projected each LiDAR point onto the output of the image semantic segmentation network, and the channel activation was linked to the intensity measurement of the

corresponding LiDAR point. Data-level fusion plays a pivotal role in enhancing the accuracy and robustness of 3D object detection. However, such methods still face many challenges, such as data misalignment, scale mismatch, and context dependence. Among these, scale inconsistency due to sensor heterogeneity remains a significant challenge. This is mainly because different sensors have different detection ranges and resolutions, for example, LiDAR can provide accurate 3D positional information at the centimeter level, whereas cameras provide 2D visual information related to pixel resolution. In this paper, we design a multi-scale feature fusion module based on graph neural networks that leverages the properties of multi-scale mechanisms to mitigate issues such as inconsistent scales.

2.2.2 Feature-Level Fusion

At the feature-level fusion [32], as shown in (a) of Figure 1, the focus shifts to the representation of features. It involves more than just concatenating the original data; instead, it encompasses weighting, fusing, or applying other processing techniques to the features obtained from different sensors in order to extract more valuable and representative feature representations. For instance, MVX-Net [15] introduced two straightforward yet effective early feature fusion methods: point fusion and voxel fusion. These methods combined RGB and point cloud features within the VoxelNet framework. ContFuse [33], on the other hand, proposed a more intricate approach that employed K-nearest neighbors (KNN), bilinear interpolation, and a learning MLP to gradually fuse the feature maps of the camera and LiDAR branches through a continuous fusion layer. UberATG-DFM [34] put forward an end-to-end learnable architecture that leveraged deep completion tasks to acquire improved multi-sensor features, achieving dense point-by-point feature fusion. PointFusion [14] introduced a global fusion and network-intensive fusion network that utilized the input 3D points as spatial anchors to predict multiple 3D box hypotheses and their corresponding confidences. The key advantage of feature-level fusion lies in its ability to leverage the strengths of different sensors fully, enabling the extraction of comprehensive and richer feature information. This, in turn, enhances the accuracy and robustness of object detection. However, feature-level fusion also encounters certain challenges. As shown in (b) of Figure 1, It necessitates the rational design of fusion strategies and weight allocation, while considering the correlation and consistency between different features. Additionally, the fusion process may lead to different levels of data redundancy due to different fusion strategies. In this paper, we propose a lightweight gating mechanism to fuse features to reduce data redundancy selectively.

2.2.3 Object-Level Fusion

Object-level fusion entails merging the object detection outcomes from various sensors to produce a comprehensive object detection result. It involves utilizing the diverse information acquired by multiple sensors, such as cameras and LiDARs, to

detect objects and consolidate them into a unified output. Fast-CLOCs [35] employed the image-based intersection over union (IoU) metric between 2D detection and projected 3D detection to measure the geometric consistency between them. This approach converted all 2D and 3D detection candidates into a harmonized joint representation, ensuring consistency across both dimensions. Liga-stereo [36] built upon the DSGN [37] model as its baseline, taking two image features from a binocular camera as input. For each assumed depth plane in the left image, the feature map of the right image is obtained through homography transformation. Subsequently, the left image features are combined with the corresponding right image features. By merging the object detection outcomes from diverse sensors, the limitations and deficiencies of individual sensors can be compensated for, leading to more comprehensive and accurate object detection results. However, object-level fusion also encounters certain challenges, such as inconsistencies in data between different sensors and the accuracy of object matching.

3 METHODOLOGY

Our objective is to devise an effective feature fusion mechanism that enhances the precision of 3D object detection.

In our approach, we introduce a novel module called the Multi-Scale Feature Fusion (MSFF) module. This module leverages a multi-scale graph neural network to extract point cloud features at various scales. Additionally, it integrates 2D image features to obtain comprehensive graph features. Simultaneously, we convert the point cloud data into a voxel representation and fuse it with the 2D image features to derive voxel features for the point cloud. Furthermore, we have developed a Discriminative Fusion Module (DFM). This module adaptively selects voxel features and graph features for fusion based on their relative importance. By doing so, we enhance the richness of the fusion features. Ultimately, this enables us to generate unified 3D features that facilitate accurate predictions of 3D object bounding boxes.

3.1 Overall Network Structure

As shown in Figure 2, the DMS3D model proposed mainly has three feature extraction branches and a feature fusion module: a 3D graph feature branch, a 2D image feature branch, a 3D voxel feature branch, and a discriminative fusion module. Firstly, the 2D image feature f^{image} is obtained by a 2D backbone network and feature pyramid (FPN) [38] network. Secondly, the point cloud data is transformed into graph structure and voxel representation respectively, and the 3D graph feature f^{graph} is obtained by fusing the feature f^{image} through the multi-scale feature fusion module, and the 3D voxel feature f^{voxel} is obtained by fusing the feature f^{image} through the voxel branch; finally, the 3D voxel feature f^{voxel} and 3D graph feature f^{graph} are input into the discriminative fusion module to obtain the final joint 3D feature f^{fusion}.

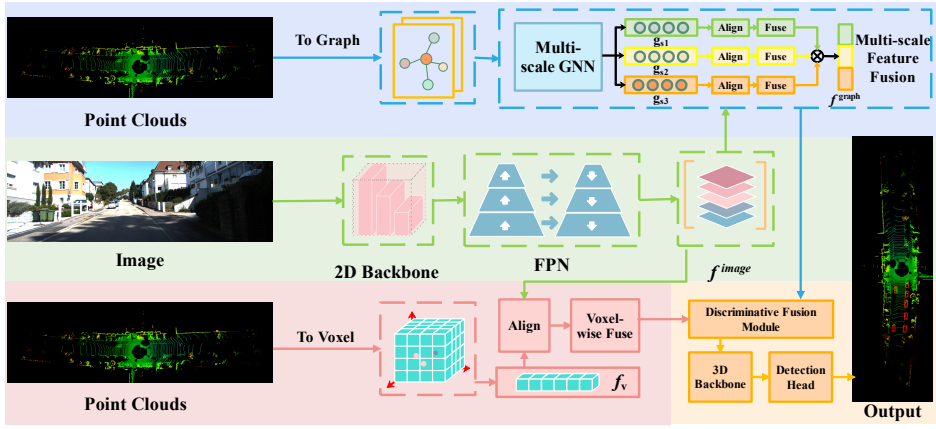


Figure 2. Overview of DMS3D architecture. Our framework mainly consists of the following parts: a) an image feature extraction network, mainly composed of a 2D backbone network and a feature pyramid network; b) a 3D graph feature branch; c) a 3D voxel feature branch; d) a discriminative fusion module; e) a 3D backbone network.

3.2 2D Image Feature

We use ResNet50 [39] network as the backbone network of 2D image feature extraction to obtain the four-layer image feature $f_{image} = [I_1, I_2, I_3, I_4]$, and then through a feature pyramid network, the five-layer image feature $f^{\text{image}} = [f_1^i, f_2^i, f_3^i, f_4^i, f_5^i]$ is finally obtained, which is obtained by adding an additional pooling layer on the basis of the original four layers, that is, the fifth layer feature is obtained by pooling operation on the basis of the top layer feature. This operation can introduce more features to further improve the detector's perception of the object.

3.3 Multi-Scale Feature Fusion Module

Point cloud data represent a collection of discrete points in space, where each point has location information and other attributes (such as color, normal vector, etc.). There is a certain connection between points, which can help the model to locate and classify objects more accurately. Different from the existing processing methods of projecting point clouds to 2D coordinates or converting them into voxel, we express this connection relationship by constructing graphs, which can not only adapt to the characteristics of close and sparse point clouds, but also provide rich context, shape and occlusion information to enrich feature representation. However, single-scale graph neural networks may not be able to adapt to targets with large scale variations and have limited understanding of the scale inconsistencies of different modal features, which may lead to inaccurate detection of distant or occluded targets. In addition, since a single-scale graph neural network cannot

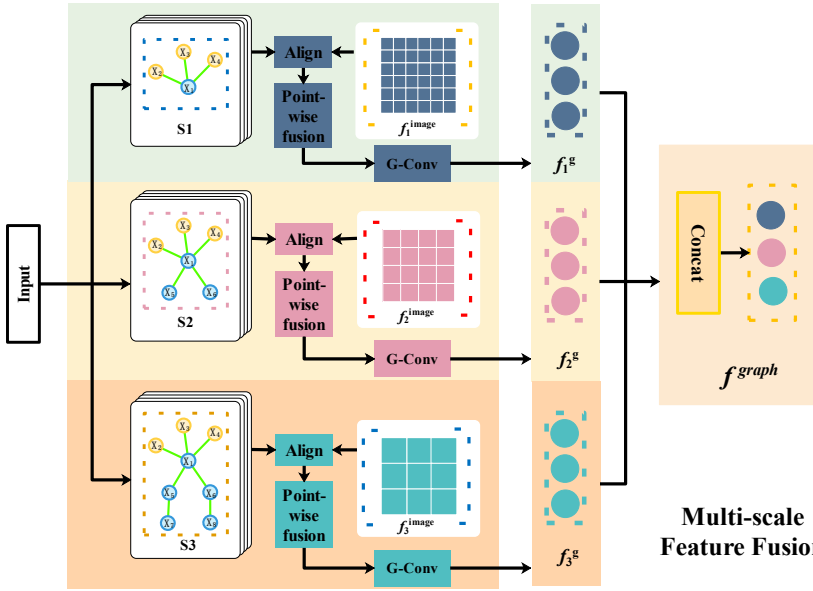


Figure 3. Multi-scale feature fusion module mainly includes point-by-point fusion and graph neural network

capture feature information at multiple scales, the model may not adequately represent key features of the target, such as shape, texture, and geometric structure, thus affecting the accuracy and robustness of object detection. To effectively alleviate the above problems, we design a multi-scale feature fusion module. Unlike existing methods that project point clouds or rotate dynamic voxels, we use multi-scale graph convolution operation to extract 3D graph features at multiple scales and fuse 2D image features at each scale to obtain richer 3D graph features. Our method has the following advantages: 1) the model is more robust and adapts to changes in targets or scenes at different scales; 2) both local structure and global context information at different scales are considered at the same time.

As shown in Figure 3, enter the original point cloud data $P_{\text{raw}} = \{p_1, p_2, p_3, \ldots, p_n\}$, where $p_i = \{(x_i, y_i, z_i, r_i) \mid i = 1, 2, \ldots, n\}$ denotes the three-dimensional coordinates of the i th point, and r denotes the reflection intensity. Firstly, the point cloud data is transformed into a graph structure $G = (V, E)$, where each point is the node V of the graph, and E represents the edge. The relationship between points is represented by an $N \times N$ adjacency matrix A, and its element a_{ij} represents the connection between node i and node j. The distance function $F(i, j)$ is used to calculate the distance between nodes, and the connection is determined according

to the threshold ϵ, as shown in Formula (1):

$$F(i,j) = \begin{cases} 1, & \text{if} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \leq \epsilon, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Subsequently, the graph convolution (GCN) [40] operation is used to construct a three-scale graph neural network. Each layer of graph convolution convolves the node features while taking into account the intricate interconnections among the nodes. Specifically, for each node i, its feature representation X_i is fused by convolution operation and adjacency matrix A to obtain the updated node feature $X_i^{'}$. This process can be expressed as:

$$X_i' = \sigma\left(\sum_{j=1}^{N} a_{ij}\left(WX_j + b\right)\right), \tag{2}$$

among them, W is a learnable weight matrix, b is a bias vector, and $\sigma(\cdot)$ is the activation function Sigmoid. Then we get the feature $S = (s_1, s_2, s_3)$ of three scales, where s_1, s_2, s_3 represent the feature set of all points in each scale, that is, $s_i = \{X_s^i\}_{i=1,2,3}$, where $X_s^i = [X_1', X_2', X_3', \ldots, X_n']_{i=1,2,3}$.

Finally, in order to be more robust when dealing with targets of different sizes, shapes, and backgrounds, we integrate multi-modal features at different scales and fully utilize their complementarity. In addition, to avoid similarity in adjacent hierarchical features, we use the first, third, and fifth layers of 2D image features and multi-scale 3D point cloud features for point-by-point fusion. Specifically, the point cloud is first mapped into a 2D image to align it, and a projection transformation function $Proj(\cdot)$ is defined. The input is the graph feature X_s^i and the transformation parameter T of each scale of the point cloud, and the output is the graph feature of each scale of the aligned point cloud:

$$X_s^{i'} = \text{Proj}\left(X_s^i, T\right), \tag{3}$$

where $X_s^{i'} = [X_1'', X_2'', X_3'', \ldots, X_n'']_{i=1,2,3}$, X_n'' denotes the position of the n^{th} point in the i^{th} scale in the 2D image coordinate system. The projection transformation function $Proj(\cdot)$ can be calculated by the following formula:

$$X_n'' = K \cdot R \cdot X_n' + T, \tag{4}$$

here, K represents the internal reference matrix of the camera, R denotes the rotation matrix of the camera, and T signifies the translation vector of the camera. Subsequently, the 2D image features are seamlessly integrated with the three-scale map features of the aligned point cloud, resulting in the acquisition of feature representations at the three different scales. This fusion process is depicted in Formula (5):

$$f_s^i = \text{Cat}\left(FC\left(X_s^{i'}\right), f^{\text{image}}\right)_{i=1,2,3}, \tag{5}$$

$FC(\cdot)$ represents the fully connected layer, which transforms the graph features into the same dimension, and $Cat(\cdot)$ is the splicing operation. Finally, the features of the three scales are spliced to obtain the 3D graph feature f^{graph}.

3.4 3D Voxel Feature Branch

While the graph neural network demonstrates efficacy in processing point cloud data, it encounters challenges when dealing with sparse edges and widely spaced points. Solely relying on a distance threshold to determine connectivity may result in indistinct boundary information and the exclusion of numerous valuable boundary points. Consequently, we propose the utilization of voxel representation for point cloud data processing. The primary rationale behind this approach is that voxelization enables the conversion of sparse boundary points into voxel grids with consistent resolution, thereby providing a unified representation of the entire point cloud data. Moreover, voxelization facilitates the generation of adjacent voxels surrounding the boundary points, thereby preserving crucial information pertaining to the vicinity of these boundaries. This technique proves instrumental in capturing object shape and boundary features, ultimately enhancing the accuracy of object detection. To this end, we use dynamic voxel feature coding (DynamicVFE) to obtain the voxel feature f_V of the point cloud, and then performs voxel-by-voxel feature fusion with the 2D image feature f^{image}. The reason is to make up for the lack of semantic information of voxel features.

Specifically, the point cloud data P_{raw} is input and voxelized to obtain the voxel feature f_V. Firstly, the size of the voxel is defined as $V_{\text{size}} = (s_x, s_y, s_z)$, which represents the width, height and depth of each voxel. The boundary of the voxel is defined as $V_{\text{boundary}} = (\min_x, \min_y, \min_z, \max_x, \max_y, \max_z)$, which represents the boundary range of the point cloud data. Then the dimension of the voxel grid is calculated, and the dimension of the voxel grid is defined as $V_{\text{dim}} = (d_x, d_y, d_z)$, which indicates how many voxels are needed in each dimension. The number of voxels in each dimension was calculated by Formula (6):

$$\begin{cases} d_x = (\max_x - \min_x)/s_x, \\ d_y = (\max_y - \min_y)/s_y, \\ d_z = (\max_z - \min_z)/s_z. \end{cases} \tag{6}$$

Create an empty voxel grid, expressed as $V_{\text{zero}} = \text{zeros}(d_x, d_y, d_z)$, where $zeros(\cdot)$ means that the value of the voxel is set to 0. Then, for each point p_i, its index in the voxel grid is calculated, as shown in Formula (7):

$$\begin{cases} idx_x = \text{floor}\left((x_i - \min_x)/s_x\right), \\ idx_y = \text{floor}\left((y_i - \min_y)/s_y\right), \quad , \\ idx_z = \text{floor}\left((z_i - \min_z)/s_z\right). \end{cases} \tag{7}$$

$floor(\cdot)$ represents the mapping from the point cloud to the voxel grid. Finally, the point p_i is assigned to the corresponding voxel grid $\text{Grid}\,(idx_x, \quad idx_y, \quad idx_z) = 1$, which indicates that there are points in the voxel, and then the voxel features $f_V = (f_{v_1}, f_{v_2}, f_{v_3}, \ldots, f_{v_m})$ are obtained.

The dynamic voxel feature encodes the feature f_V, which is then aligned with the 2D image feature f^{image}, resulting in the generation of feature representations that are enriched with semantic information. Through the alignment of multimodal data, various modalities of information are integrated, filling in gaps and ensuring a consistent representation of the data. This alignment process contributes to enhanced task performance, enabling more accurate and comprehensive data analysis and decision-making. Ultimately, it provides valuable support for precise and comprehensive data analysis and decision-making processes. Therefore, we first project each voxel f_{v_i} $(i = 1, 2, 3, \ldots, m)$ onto the image plane, and calculate the pixel coordinates of the voxel center point on the image plane:

$$(U_{v_i}, V_{v_i}) = \frac{(K \cdot T \cdot V_{v_i})}{(K \cdot T \cdot P_{v_i})}, \tag{8}$$

where K is the internal reference matrix of the camera, T is the external reference matrix of the camera, V_{v_i} is the coordinates of the center point of the voxel in the camera coordinate system, and P_{v_i} is the coordinates of the center point of the voxel in the world coordinate system. The voxel feature f_{v_i} is assigned to the image feature closest to the projection position to align and return the aligned feature f_{align}. Then f_{align} and f^{image} are fused to obtain the 3D voxel feature f^{voxel}, as shown in Formula (9):

$$f^{\text{voxel}} = \text{Cat}\left(\text{Trans}\left(f^{\text{image}}\right), f_{\text{align}}\right), \tag{9}$$

$Trans(\cdot)$ represents the operation of converting features into the same dimension, and $Cat(\cdot)$ represents the feature splicing operation.

3.5 Discriminative Fusion Module

Inspired by [41], the data of different modalities have different feature information, such as the 3D graph feature f^{graph} and the 3D voxel feature f^{voxel} obtained in the above process. The former contains rich context and multi-scale information, but there are some deficiencies in the extraction of edge information; although the latter is not as rich as the feature f^{graph}, it can capture the shape and boundary features of the object, which is beneficial to complement the 3D graph features to a certain extent. In addition, if the 3D graph feature f^{graph} and the 3D voxel feature f^{voxel} are directly fused, it may cause information redundancy and increase the amount of calculation. Consequently, we have devised a discriminative fusion module with the objective of selectively merging features using a lightweight gating mechanism. Different from existing approaches that utilize attention mechanisms or direct concatenation, we develop a lightweight gating mechanism to se-

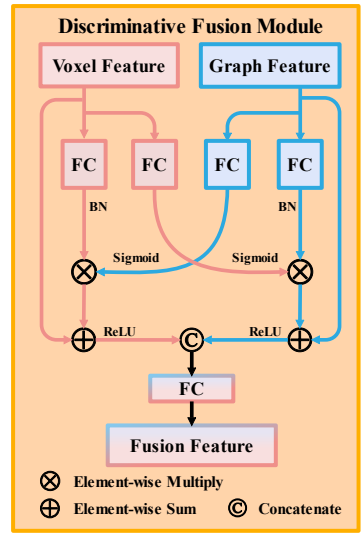


Figure 4. Explanation of the discriminative fusion module

lectively fuse features from different modalities. Our method offers several advantages:

1. It reduces computational overhead;
2. It enhances the adaptability of the model to diverse modal features, making the network more flexible.

By incorporating this module, we achieve an improved ability to discern relevant features and enhance the overall robustness of the model.

As shown in Figure 4, input the 3D graph feature f^{graph} and the 3D voxel feature f^{voxel}, and use a lightweight gating unit to control the weight of feature fusion. Firstly, a gating variable is defined to represent the weight of the selected graph structure features or voxel features. Let $W = \{w_i\}$ be the gated variable, where w_i is a value between 0 and 1. Subsequently, a fully connected layer is employed, followed by the utilization of a Sigmoid function as the activation function. This configuration enables the generation of gating variables by taking the graph structure features and voxel features as inputs. The specific formula can be expressed as:

$$w_i = \text{Sigmoid}\left(W^T \cdot f^{\text{multiply}}\right), \tag{10}$$

among them, W is the parameter of the gating unit, and $f^{multiply} = Multiply(f^{graph}, f^{voxel})$ is a new feature vector formed by multiplying graph structure features and voxel features. At the same time, the two features obtained by adding and splicing $f^{multiply}$ with f^{graph} or f^{voxel} are passed through a fully connected layer to obtain

the joint feature f^{fusion} finally used for detection, as shown in Formula (11):

$$f^{\text{fusion}} = \text{Cat}\left(\text{Sum}\left(f^{\text{multiply}}, f^{\text{graph}}\right), \text{Sum}\left(f^{\text{multiply}}, f^{\text{voxel}}\right)\right), \tag{11}$$

where $Sum(\cdot)$ denotes the sum of features. Finally, we use a 3D detection head with an anchor frame for detection. The ground truth frame and anchor points are defined by $(x, y, z, w, l, h, \theta)$. The localized regression residual between the ground true value and the anchor point is defined as:

$$\begin{aligned} &\Delta x = \tfrac{x^{gt}-x^a}{d^a}, && \Delta y = \tfrac{y^{gt}-y^a}{d^a}, && \Delta z = \tfrac{z^{gt}-z^a}{h^a}, \\ &\Delta w = \log \tfrac{w^{gt}}{w^a}, && \Delta l = \log \tfrac{l^{gt}}{l^a}, && \Delta h = \log \tfrac{h^{gt}}{h^a}, \\ &\Delta \theta = \sin\left(\theta^{gt} - \theta^a\right), \end{aligned} \tag{12}$$

where x^{gt} and x^a are the ground truth value and anchor frame respectively, $d^a = \sqrt{(w^a)^2 + (l^a)^2}$.

3.6 Loss Function

We utilize the multi-task loss function for jointly optimizing the model. The total loss is:

$$L = \frac{1}{N_{\text{pos}}}\left(\beta_{loc} L_{loc} + \beta_{cls} L_{cls} + \beta_{dir} L_{dir}\right), \tag{13}$$

where N_{pos} is the number of positive anchor points, L_{loc} is the regression loss for location and dimension, L_{cls} is the classified loss and L_{dir} is the direction classification loss, $\beta_{loc} = 2$, $\beta_{cls} = 1$, $\beta_{dir} = 0.2$ are the constant coefficients of our loss formula.

The overall localization loss is:

$$L_{loc} = \sum_{b \in (x,y,z,w,l,h,\theta)} \text{SmoothL1}(\Delta b), \tag{14}$$

since the angle localization loss cannot distinguish flipped bounding boxes, the softmax classification loss L_{dir} is used to learn in the discrete direction, which same to SECOND [6]. Object classification loss using focal loss:

$$L_{cls} = -\alpha_a \left(1 - p^a\right)^r \log p^a, \tag{15}$$

where p^a denotes the class probability for an anchor point. We use the original setting with $\alpha = 0.25$ and $\gamma = 2$.

The loss function is optimized with AdamW, and the initial learning rate is $1 \times e^{-4}$, decaying by 0.1 times every 15 cycles. The number of epochs is 40, and the batch size is 2.

4 EXPERIMENT

4.1 Datasets and Evaluation Metrics

KITTI Dataset: All experiments use the KITTI object detection benchmark data set [42], which consists of samples with both LiDAR point clouds and images. These samples were initially divided into 7 481 training samples and 7 518 test samples. There are three different levels: simple, medium, and difficult, which are determined based on the size of the object, visibility (occlusion), and truncation. Average precision (AP) is calculated using 40 recalls as a validation metric. For cars, pedestrians and cyclists, the IoU thresholds are 0.7, 0.5 and 0.5, respectively. We divide the official training set into 3 712 training samples and 3 769 test samples. At the time of test submission, we train on the training set containing 7 481 training samples, test on the test set containing 7 518 samples, and generate submission documents to upload to the official website for verification.

nuScenes Dataset: The nuScenes dataset [43] consists of 700, 150, and 150 scenes for training, validation, and testing, respectively. In each frame, there is one point cloud data along with six calibrated images, covering a 360-degree horizontal field of view. When it comes to 3D object detection, the primary evaluation metrics used are the mean Average Precision (mAP) and the nuScenes detection score (NDS). Notably, the mAP is calculated based on the Bird's Eye View (BEV) center distance instead of the 3D Intersection over Union (IoU). The final mAP is determined by averaging the results over different distance thresholds, specifically 0.5 m, 1 m, 2 m, and 4 m, across ten object classes. NDS serves as a comprehensive metric that encompasses mAP as well as various attribute metrics such as translation, scale, orientation, velocity, and other box attributes.

4.2 Implementation Details

4.2.1 2D Backbone

The pre-trained ResNet50 is used as the backbone network for image feature extraction. In the feature pyramid stage, the number of input channels of each layer is set to 256, 512, 1 024, 2 048, the number of output channels is set to 256, and the output is 5 layers.

4.2.2 Multi-Scale Graph Neural Network

We use three scales of graph convolution. The convolution kernel is set to ((32, 32), (64, 64), (128,)). The last layer is completed by a graph convolution operation, and the other two layers are completed by two graph convolution layers. The distance

threshold is set to 0.1 meters, the point-by-point fusion module embedded in each scale is consistent with that proposed in [15], and the output channel is set to 128.

4.2.3 Voxel Feature Encoding

Four dimensions (x, y, z, r) of 17 600 points were sampled as input. The range of the point cloud was $[0, -40, -3, 70.4, 40, 1]$. The point cloud data was encoded by dynamic voxel feature coding, and the voxel size was set to $[0.05, 0.05, 0.1]$. The voxel-by-voxel fusion layer is embedded, and the feature map of the five-layer image is used for fusion. The image channel is 256, the point cloud channel is 64, and the output channel is 128.

4.2.4 Discriminative Fusion Module

Sigmoid is used as the activation function of gated multiplication, and ReLU is used as the activation function of gated addition. The weight is initialized by normal distribution. In order to prevent overfitting, the Dropout layer is set to 0.2, and the regularization parameter is set to 0.01.

4.2.5 Other Settings

The 3D backbone network is consistent with [6]. The 3D head of the detected head anchor, the feature channel is set to 512, the classification loss function is FocalLoss, the bounding box loss is SmoothL1Loss, the direction loss is CrossEntropyLoss, and the Sigmoid activation function is not used.

4.3 Main Results

Our comparative data is derived from the official websites of KITTI and nuScenes, supplemented by corresponding research papers. In the provided results in Table 1, '-' indicates that the algorithm did not conduct experiments for the respective metrics. 'L' represents LiDAR, 'R' represents RGB image. Additionally, due to challenges in obtaining source code, reproducing results for certain algorithms was difficult. Even for algorithms that could be implemented, variations may occur due to differences in servers. The primary benchmark for comparison on the KITTI dataset is the test set.

4.3.1 Results on the KITTI Test Set

We follow the standard KITTI evaluation scheme (IoU = 0.7) to measure the detection performance, and trains on the LiDAR point cloud and image, and compares it with the fusion method using LiDAR and image and the method using only LiDAR point cloud. Tables 1 and 2 show the mean average precision (mAP) scores of DMS3D compared to the advanced KITTI test set methods evaluated using 3D and

Method	Modality	Car (3D %)				Pedestrian (3D %)				Cyclist (3D %)			
		Esay	Moderate	Hard	mAP	Esay	Moderate	Hard	mAP	Esay	Moderate	Hard	mAP
StereoDistill [44]	L	81.66	66.39	57.39	68.48	44.12	32.23	28.95	35.10	63.96	44.02	39.19	49.06
WS3D [45]	L	80.99	70.59	64.23	71.94	–	–	–	–	–	–	–	–
DSGN++ [46]	L	83.21	67.37	59.91	70.16	43.05	32.74	29.54	35.11	62.82	43.90	39.21	48.64
Pseudo-LiDAR++ [47]	L	68.38	54.88	49.16	57.47	–	–	–	–	–	–	–	–
SeSame [28]	L	81.51	75.05	70.53	75.70	46.53	37.37	33.56	39.15	70.97	54.36	48.66	57.99
Pass-PointPillars [48]	L	84.72	74.85	69.05	76.21	–	–	–	–	–	–	–	–
MV3D [12]	L+R	74.97	63.63	54.00	64.20	–	–	–	–	–	–	–	–
AVOD [13]	L+R	76.39	66.47	60.23	67.70	36.10	27.86	25.76	29.91	57.19	42.08	38.29	45.85
ESGN [49]	L+R	65.80	46.39	38.42	50.20	14.05	10.27	9.02	11.11	13.84	7.69	6.75	9.43
PI-RCNN [18]	L+R	84.37	74.82	70.03	76.41	–	–	–	–	–	–	–	–
DMS3D (Our)	L+R	**85.17**	**75.20**	**71.18**	**77.18**	**47.88**	**40.62**	**37.48**	**43.87**	**80.49**	**63.52**	**56.78**	**66.93**

Table 1. Comparison of 3D average precision (AP) performance on KITTI test set evaluated with recall 40 positions. Results are evaluated on the online KITTI test server.

Method	Modality	Car (BEV %)				Pedestrian (BEV %)				Cyclist (BEV %)			
		Esay	Moderate	Hard	mAP	Esay	Moderate	Hard	mAP	Esay	Moderate	Hard	mAP
StereoDistill [44]	L	89.03	78.59	69.34	79.10	50.79	37.75	34.28	40.94	69.46	48.37	42.69	53.51
WS3D [45]	L	90.96	84.93	77.96	84.62	–	–	–	–	–	–	–	–
DSGN++ [46]	L	88.55	78.94	69.74	79.08	50.26	38.92	35.12	41.43	68.29	49.37	43.79	53.82
Pseudo-LiDAR++ [47]	L	84.61	73.80	65.59	74.67	–	–	–	–	–	–	–	–
SeSame [28]	L	89.86	85.62	80.95	85.48	50.12	41.59	37.79	43.17	76.95	59.36	53.14	63.15
Pass-PointPillars [48]	L	91.07	**87.23**	81.98	**86.76**	–	–	–	–	–	–	–	–
MV3D [12]	L+R	86.62	78.93	69.80	78.45	–	–	–	–	–	–	–	–
AVOD [13]	L+R	89.75	84.95	78.32	84.34	42.58	33.57	30.14	35.43	64.11	48.15	42.37	51.54
ESGN [49]	L+R	78.10	58.12	49.28	61.80	17.94	13.03	11.54	14.17	15.78	9.02	7.96	10.92
PI-RCNN [18]	L+R	**91.44**	85.81	81.00	86.08	–	–	–	–	–	–	–	–
DMS3D (Our)	L+R	90.40	86.26	**82.00**	86.22	**53.48**	**45.82**	**42.70**	**47.33**	**84.87**	**69.22**	**62.38**	**72.16**

Table 2. Comparison of BEV average precision (AP) performance on KITTI test set evaluated with recall 40 positions. Results are evaluated on the online KITTI test server.

bird's-eye view (BEV), respectively. Our method has a 0.35 %, 3.25 % and 9.16 % improvement on the 3D AP medium benchmark of cars, pedestrians and cyclists, respectively, and a 0.81 % improvement on 3D mAP. On the BEV AP benchmark, there are 8.49 % and 9.52 % increases in the medium tests of pedestrians and cyclists, respectively. Under the difficult standards of automobiles, the performance has improved by 0.02 % compared to Pass-PointPillars [48]. In addition, there is also a 4.64 % increase in the BEV mAP compared to SeSame [28]. The performance improvement primarily stems from the design and implementation of multi-scale feature fusion and the discriminative fusion module. It effectively alleviates the issue of inconsistent scales in multimodal data and obtains more comprehensive features, optimizing the model.

4.3.2 Results on the KITTI Validation Set

Tables 3 and 4 show three different categories of 3D performance and BEV performance on the KITTI verification set, and compare them with the fusion method using LiDAR and image and the method using only LiDAR point cloud. Our method has certain advantages in pedestrian detection in 3D object detection, and has achieved an increase of 16.77 % compared with other data in the table on the medium standard. In BEV detection, our method also shows excellent performance for pedestrian detection, achieving a 15.96 % improvement on the medium standard. Encouraging results have been achieved on the KITTI verification set, and the experimental results further prove the effectiveness of the method.

4.3.3 Results on the nuScenes Val Set

As shown in Table 5, in order to test the generalization performance of the model, we tested our model on the nuScenes validation set. Compared to BEVFusion, our method has improved mAP by 2.6 % and NDS by 2.1 %. This verifies that our model has a certain degree of generalization performance.

4.4 Ablation Experiment

We conducted experiments on the KITTI test set to test the hyperparameters, and verified each component of the proposed method on the 3D benchmark and the BEV benchmark, respectively.

4.4.1 Single-Scale Graph Neural Network (SSGNN)

In order to verify the effectiveness of the graph neural network in processing point cloud data, we set the single voxel processing and 2D image feature fusion as the baseline. It can be seen from the first and second rows of Table 6 and Table 7 that the car category does not improve on the 3D AP benchmark, but it has 1.76 % and 11.01 % improvement on the other two categories, respectively. At the same time,

Method	Modality	Car (3D %)				Pedestrian (3D %)				Cyclist (3D %)			
		Esay	Moderate	Hard	mAP	Esay	Moderate	Hard	mAP	Esay	Moderate	Hard	mAP
PointRCNN [24]	L	86.96	75.64	70.70	77.77	47.98	39.37	36.01	41.12	74.96	58.82	52.53	62.10
Part-A^2 [50]	L	87.81	78.49	73.51	79.94	53.10	43.35	40.06	45.50	79.17	63.52	56.93	66.54
TANet [51]	L	83.81	75.38	67.66	75.62	–	–	–	–	–	–	–	–
DSGN [37]	L	73.50	52.18	45.14	56.94	20.53	15.55	14.15	16.74	27.76	18.17	16.21	20.71
LIGA-stereo [36]	L	81.39	64.66	57.22	67.76	40.46	30.00	27.07	32.51	54.44	36.86	32.06	41.12
SVGA-Net [52]	L	87.33	80.47	75.91	81.24	–	–	–	–	–	–	–	–
StereoDistill [44]	L	87.57	69.75	62.92	73.41	55.19	46.76	40.42	47.46	69.43	42.31	39.10	50.28
MV3D [12]	L+R	71.29	62.68	56.56	63.51	–	–	–	–	–	–	–	–
F-PointNet [29]	L+R	82.19	69.79	60.59	70.86	50.53	42.15	38.08	43.59	72.27	56.12	49.01	59.13
UberATG-MMF [34]	L+R	88.40	77.43	70.22	78.68	–	–	–	–	–	–	–	–
RangeIoUDet [53]	L+R	88.60	79.80	76.76	81.72	–	–	–	–	83.12	67.77	60.26	70.38
HMFI [54]	L+R	**88.90**	**81.93**	**77.30**	**82.71**	50.88	42.65	39.78	44.44	**84.02**	**70.37**	**62.57**	**72.32**
CL3D [16]	L+R	87.45	80.28	76.21	81.31	47.30	39.42	36.97	41.23	77.33	62.02	55.52	64.96
DMS3D (Our)	L+R	87.95	78.38	75.86	80.73	**65.88**	**59.42**	**53.33**	**59.54**	82.21	61.92	57.69	67.27

Table 3. Comparison of 3D average precision (AP) performance on recall 40 positions of the KITTI validation set. Results are evaluated using the official test set split provided by KITTI.

Method	Modality	Car (BEV %)				Pedestrian (BEV %)				Cyclist (BEV %)			
		Esay	Moderate	Hard	mAP	Esay	Moderate	Hard	mAP	Esay	Moderate	Hard	mAP
PointRCNN [24]	L	92.13	87.39	82.72	87.41	54.77	46.13	42.84	47.91	82.56	67.24	60.28	70.03
Part-A^2 [50]	L	91.70	87.79	84.61	80.03	59.04	49.81	45.92	51.59	83.43	68.73	61.85	71.34
TANet [51]	L	91.58	86.54	81.19	86.44	–	–	–	–	–	–	–	–
DSGN [37]	L	82.90	65.05	56.60	68.18	26.61	20.75	18.86	22.07	31.23	21.04	18.93	23.73
LIGA-stereo [36]	L	88.15	76.78	67.40	77.44	44.71	34.13	30.42	36.42	58.95	40.60	35.27	44.94
SVGA-Net [52]	L	92.07	**89.88**	**85.59**	**89.18**	–	–	–	–	–	–	–	–
StereoDistill [44]	L	–	–	–	–	–	–	–	–	–	–	–	–
MV3D [12]	L+R	86.62	78.93	69.80	78.45	–	–	–	–	–	–	–	–
F-PointNet [29]	L+R	91.17	84.67	74.77	83.54	57.13	49.57	45.48	50.73	77.26	61.37	53.78	64.14
UberATG-MMF [34]	L+R	**93.67**	88.21	81.99	87.96	–	–	–	–	–	–	–	–
RangeIoUDet [53]	L+R	92.28	88.59	85.83	88.90	–	–	–	–	**85.99**	**71.49**	**63.62**	**73.70**
HMFI [54]	L+R	–	–	–	–	–	–	–	–	–	–	–	–
CL3D [16]	L+R	–	–	–	–	–	–	–	–	–	–	–	–
DMS3D (Our)	L+R	92.23	87.46	85.39	88.36	**72.34**	**65.77**	**60.21**	**66.11**	84.47	66.32	62.19	70.99

Table 4. Comparison of BEV average precision (AP) performance on recall 40 positions of the KITTI validation set. Results are evaluated using the official test set split provided by KITTI.

Methods	Modality	mAP	NDS
TransFusion-L [55]	L	60.0	66.8
FocalsConv [56]	L	61.2	68.1
CenterPoint [9]	L	59.6	66.8
FUTR3D [57]	L + C	64.2	68.0
UVTR [58]	L + C	65.4	70.2
TransFusion [55]	L + C	67.5	71.3
BEVFusion [17]	L + C	68.5	71.4
DMS3D (Our)	L + C	**71.1**	**73.5**

Table 5. Comparison of detection results on nuScenes validation set. L represents LiDAR and C represents Camera.

Component					3D (%)			
Baseline	SSGNN	MSGNN	MSFF	DFM	Car	Ped.	Cyc.	mAP
✓					72.34	36.39	46.42	51.72
✓	✓				72.34	38.15	57.43	55.97
✓		✓			72.61	38.35	61.78	57.58
✓			✓		73.22	39.62	62.39	58.41
✓				✓	75.20	40.62	63.52	59.78

Table 6. Comparison of 3D average precision (AP) performance on KITTI test set evaluated using online KITTI test server, calculated with recall 40 positions. Ped. denotes the pedestrian class and Cyc. denotes the cyclist class.

it has increased by 0.73 %, 1.35 % and 12.77 % on the BEV AP benchmark, respectively. This demonstrates that, compared to the single voxel processing method, incorporating graph-based representations significantly enhances feature expressiveness by capturing spatial relationships between neighboring points. Unlike voxel-based methods that treat points as independent units within discrete grid cells, the graph structure enables direct information propagation across points, improving contextual understanding. This enhanced point-to-point correlation allows the model to better preserve object structures, distinguish between similar objects, and improve detection performance in complex environments.

Component					3D (%)			
Baseline	SSGNN	MSGNN	MSFF	DFM	Car	Ped.	Cyc.	mAP
✓					85.09	42.32	51.97	59.79
✓	✓				85.82	43.67	64.74	64.74
✓		✓			84.73	44.54	69.43	66.23
✓			✓		86.85	45.29	69.05	67.06
✓				✓	86.25	45.82	69.22	67.10

Table 7. Comparison of BEV average precision (AP) performance on KITTI test set evaluated using online KITTI test server, calculated with recall 40 positions. Ped. denotes the pedestrian class and Cyc. denotes the cyclist class.

4.4.2 Multi-Scale Graph Neural Network (MSGNN)

Based on the single-scale graph neural network, we have improved it and added two scales of features. Compared with the single-scale graph neural network on the 3D AP benchmark, it can be seen from the second and third rows of Table 6 and Table 7 that although there is a slight improvement in the car and the human, the performance is improved by 4.35 % in the cyclist detection, and the average accuracy is also improved by 1.61 %. In addition, on the BEV AP benchmark, there were 0.87 % and 4.69 % increases for pedestrians and cyclists, respectively, and the overall mAP increased by 1.49 %. This shows that the multi-scale feature extraction method is more conducive to adapting to the object detection of multiple scenes and complex structures than the single scale.

4.4.3 Multi-Scale Feature Fusion (MSFF)

In order to overcome the lack of texture and other information in the graph structure features, we add the features of 2D images to the graph features of each scale, and fuses the features of three scales. It can be seen from the third and fourth lines of Table 6 that on the basis of 3D detection, this method increases by 0.61 %, 1.27 % and 0.61 % respectively compared with the multi-scale graph neural network in the three categories of automobile, pedestrian and cyclist. The mAP increased by 0.83 % and increased by 6.69 % compared with the baseline. In addition, the performance of the three categories is improved by 2.12 %, 0.75 % and 0.61 % respectively on the BEV detection benchmark, and the mAP is improved by 0.83 %. These improvements result from 2D images providing rich semantic information (e.g., texture and color) that graph-based point cloud features lack. By aligning and fusing 2D and multi-scale graph features, the model enhances both local details and global structure, leading to better object recognition and detection accuracy.

4.4.4 Discriminative Fusion Module (DFM)

In order to reduce the redundancy of feature information and reduce the interference of noise, we design a lightweight discriminative fusion module, which selectively fuses features. On the 3D AP benchmark, compared with the direct fusion feature method, the proposed method improves the detection accuracy of cars, pedestrians and cyclists by 1.98 %, 1.0 % and 1.13 %, and the overall mAP is increased by 1.37 %. Compared with the baseline, the car, pedestrian and cyclist categories are increased by 1.86 %, 4.23 % and 12.49 %, respectively, and the mAP is increased by 7.73 %. In addition, on the BEV AP benchmark, compared with the direct fusion method, the performance is improved by 0.53 % and 0.17 % respectively when detecting pedestrians and cyclists. In addition, compared with the baseline, the accuracy of the three categories increased by 1.16 %, 3.5 % and 17.25 %, respectively, and the mAP increased by 7.31 %. The experimental results show that the selective fusion of different modal features is beneficial to reducing the redundancy of information to

improve the accuracy of detection, and further prove that our proposed method has certain advantages.

4.4.5 Train Time

Component	FPS (img/s)	time (iter/s)			
		Slowest	Fastest	Average	std
DMS3D (without DFM)	7.8	0.6221	0.6018	0.6155	0.0217
DMS3D (with CA)	8.9	0.5431	0.5123	0.5209	**0.0163**
DMS3D	**10.5**	**0.5042**	**0.4931**	**0.4987**	0.0182

Table 8. Slowest represents the average training time for each iteration of the slowest epoch, Fastest represents the average training time for each iteration of the fastest epoch. Average represents the average training time for each iteration of all epochs, and std represents the standard deviation of all training time.

Component	time (iter/s)			
	Slowest	Fastest	Average	std
Baseline	1.2024	1.1909	1.1959	**0.0028**
MSFF	0.5635	0.5114	0.5319	0.0143
DFM	**0.5440**	**0.5058**	**0.5179**	0.0079

Table 9. Slowest represents the average training time for each iteration of the slowest epoch, Fastest represents the average training time for each iteration of the fastest epoch. Average represents the average training time for each iteration of all epochs, and std represents the standard deviation of all training time.

In order to verify the effectiveness of the discriminative fusion module in solving the problem of long model training time caused by feature redundancy in multi-mode feature fusion, we compared the benchmark model, multi-scale feature fusion (MSFF) and discriminative fusion (DFM) three models, in which DFM is the final model algorithm. The experimental results in Table 8 indicate the effectiveness of our designed DFM module compared to other fusion methods that do not select features, such as using cross attention fusion or direct fusion. Our designed model has improved FPS metrics by 2.7 and 1.6 compared to the other two methods, respectively. In addition, it has shortened the average iteration time per epoch by 0.1168 s and 0.0222 s. Table 9 shows the results of the comparison. It can be seen that the discriminative fusion module we designed has a significant change in the improvement of training time. The average training time is reduced by 0.014 s, and it can be seen from the standard deviation that the distribution of training time is more balanced and the model is more stable. The shortened training time is a direct result of the DFM selectively fusing only the most discriminative features, reducing the amount of redundant information that needs to be processed. By focusing on the most relevant features, the model avoids unnecessary computations,

leading to faster convergence during training. Additionally, the lightweight gating mechanism in DFM streamlines the fusion process by efficiently controlling the flow of information, further minimizing computational overhead and improving training efficiency. As a result, this not only accelerates the model's training time but also enhances its stability, as reflected in the more balanced distribution of training times across epochs.

4.4.6 Component Analysis

	3D (%)				
Component	Car	Ped.	Cyc.	mAP	times (s)
SSGNN*	71.95	38.21	56.37	55.51	0.8734
SSGNN	72.34	38.15	57.43	55.97	0.9352
MSGNN*	72.18	38.19	58.46	56.28	**0.6812**
MSGNN	**72.61**	**38.35**	**61.78**	**57.58**	0.7903

Table 10. Effect of 2D image features on module performance. "*" indicates that no 2D image features are added.

Table 10 shows the effect of 2D image features on the performance of the multi-scale feature fusion module under the 3D AP benchmark. The first and second rows in Table 10 indicate that adding 2D image features increases the detection performance by 0.39 % and 1.06 % for cars and pedestrians, respectively, under the single scale setting. The fourth and third rows of the table show that under the multi-scale setting, the accuracy increases by 0.43 %, 0.16 % and 3.32 % for the three categories, respectively. This indicates the effectiveness of adding 2D image features.

4.5 Qualitative Results

Figure 5 shows the qualitative results of several comprehensive scenarios from the KITTI test set. As shown in the figure, the DMS3D proposed can accurately detect objects even in challenging intersections or complex scenes containing pedestrians and cyclists. Figure 6 shows the visual detection results of our model on the nuScenes dataset. It can be observed that our model can still accurately detect targets and maintain stable recognition performance in different scenarios and complex environments. These qualitative examples further provide intuitive evidence for the effectiveness of our method.

5 CONCLUSION

We are committed to the 3D object detection task and propose a new multi-scale and discriminative fusion module (DMS3D). The utilization of a multi-scale graph

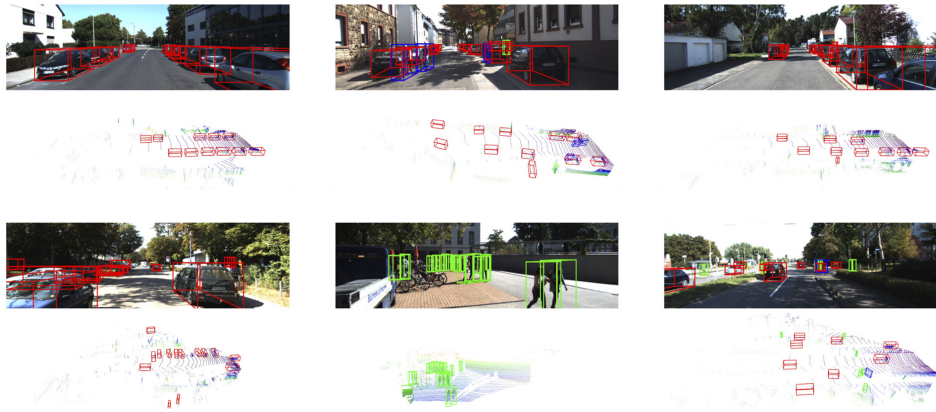


Figure 5. Here are the qualitative results of the test set of the KITTI dataset. Six complex scenes containing cars, pedestrians, and cyclists are shown. The red bounding box represents the car class detected by the model, the green bounding box represents the pedestrian class detected by the model, and the blue bounding box represents the cyclist class detected by the model.

neural network allows for the fusion of 2D image features, resulting in point cloud features that encompass a wealth of contextual and semantic information. To augment the graph features, the voxel representation is incorporated. Furthermore, a lightweight gating mechanism is devised to selectively merge the graph and voxel features, thereby reducing superfluous data and enhancing the model's detection capabilities. Evaluations conducted on the KITTI dataset demonstrate the superiority of our proposed method compared to certain existing multi-modal 3D object detection approaches.

Nevertheless, it should be noted that DMS3D exhibits suboptimal detection performance in complex scenarios characterized by severe occlusion and objects with similar shapes and colors. This can be attributed to the presence of noise and misalignment in the multi-modal features, which significantly impairs detection accuracy. To address such issues, we plan to optimize our approach in future work through methods such as multimodal data cleaning and reinforcement learning. Additionally, we prioritize the development of lightweight modules to facilitate efficient multi-modal 3D detection tasks.

Acknowledgements

The key research project of colleges and universities in Henan Province of China (Grant No. 23A520055), Henan Provincial Key Laboratory of Ecological Environment Protection and Restoration of the Yellow River Basin Open Research Fund (Grant No. LYBEPR202202).

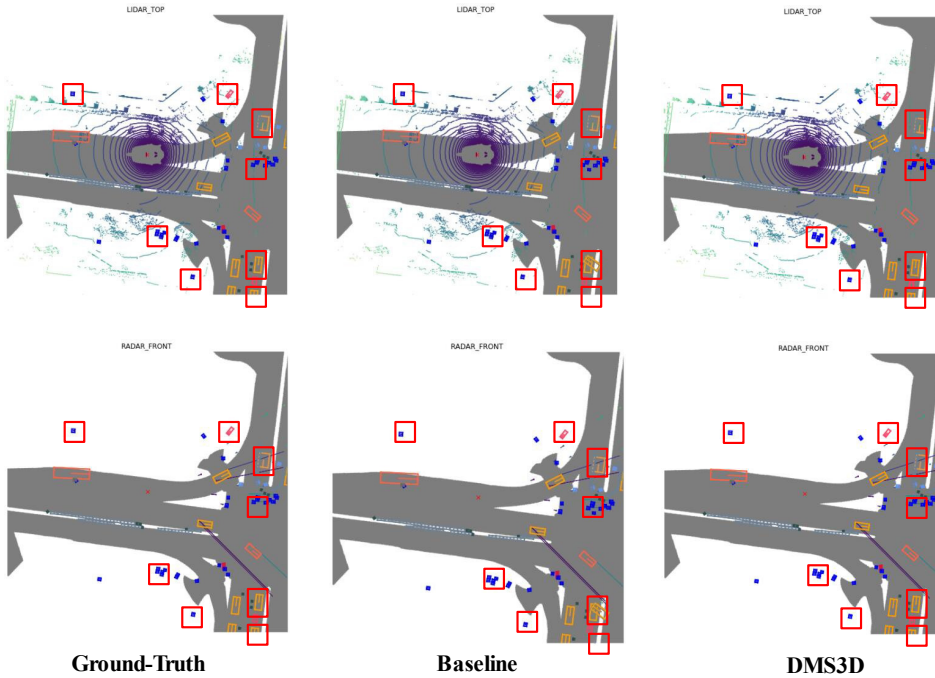


Figure 6. Here are the qualitative results of the test set of the nuScenes dataset. From top to bottom, it is divided into top level LiDAR perspective and bottom level millimeter wave radar perspective. The color of the car results is yellow, pedestrians are blue, and bicycles are pink.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that can have appeared to influence the work reported in this paper.

REFERENCES

[1] Yin, J.—Luo, D.—Yan, F.—Zhuang, Y.: A Novel Lidar-Assisted Monocular Visual SLAM Framework for Mobile Robots in Outdoor Environments. IEEE Transactions on Instrumentation and Measurement, Vol. 71, 2022, pp. 1–11, doi: 10.1109/TIM.2022.3190031.

[2] He, G.—Zhang, Q.—Zhuang, Y.: Online Semantic-Assisted Topological Map Building with LiDAR in Large-Scale Outdoor Environments: Toward Robust Place Recognition. IEEE Transactions on Instrumentation and Measurement, Vol. 71, 2022, pp. 1–12, doi: 10.1109/TIM.2022.3191648.

[3] Muhammad, K.—Ullah, A.—Lloret, J.—Del Ser, J.—de Albuquerque, V. H. C.: Deep Learning for Safe Autonomous Driving: Current Challenges and Future Directions. IEEE Transactions on Intelligent Transportation Systems, Vol. 22, 2021, No. 7, pp. 4316–4336, doi: 10.1109/TITS.2020.3032227.

[4] Zhou, Y.—Tuzel, O.: VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 4490–4499, doi: 10.1109/CVPR.2018.00472.

[5] Yan, C.—Salman, E.: Mono3D: Open Source Cell Library for Monolithic 3-D Integrated Circuits. IEEE Transactions on Circuits and Systems I: Regular Papers, Vol. 65, 2018, No. 3, pp. 1075–1085, doi: 10.1109/TCSI.2017.2768330.

[6] Yan, Y.—Mao, Y.—Li, B.: SECOND: Sparsely Embedded Convolutional Detection. Sensors, Vol. 18, 2018, No. 10, Art. No. 3337, doi: 10.3390/s18103337.

[7] Shi, X.—Ye, Q.—Chen, X.—Chen, C.—Chen, Z.—Kim, T. K.: Geometry-Based Distance Decomposition for Monocular 3D Object Detection. 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 15172–15181, doi: 10.1109/ICCV48922.2021.01489.

[8] Reading, C.—Harakeh, A.—Chae, J.—Waslander, S. L.: Categorical Depth Distribution Network for Monocular 3D Object Detection. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 8551–8560, doi: 10.1109/CVPR46437.2021.00845.

[9] Yin, T.—Zhou, X.—Krähenbühl, P.: Center-Based 3D Object Detection and Tracking. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 11779–11788, doi: 10.1109/CVPR46437.2021.01161.

[10] Qi, C. R.—Su, H.—Mo, K.—Guibas, L. J.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 77–85, doi: 10.1109/CVPR.2017.16.

[11] Qi, C. R.—Yi, L.—Su, H.—Guibas, L. J.: PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In: Guyon, I., Von Luxburg, U., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.): Advances in Neural Information Processing Systems 30 (NIPS 2017). Curran Associates, Inc., 2023, pp. 5099–5108, https://proceedings.neurips.cc/paper_files/paper/2017/file/d8bf84be3800d12f74d8b05e9b89836f-Paper.pdf.

[12] Chen, X.—Ma, H.—Wan, J.—Li, B.—Xia, T.: Multi-View 3D Object Detection Network for Autonomous Driving. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6526–6534, doi: 10.1109/CVPR.2017.691.

[13] Ku, J.—Mozifian, M.—Lee, J.—Harakeh, A.—Waslander, S. L.: Joint 3D Proposal Generation and Object Detection from View Aggregation. 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018, pp. 1–8, doi: 10.1109/IROS.2018.8594049.

[14] Xu, D.—Anguelov, D.—Jain, A.: PointFusion: Deep Sensor Fusion for 3D Bounding Box Estimation. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 244–253, doi: 10.1109/CVPR.2018.00033.

[15] Sindagi, V. A.—Zhou, Y.—Tuzel, O.: MVX-Net: Multimodal Voxelnet for 3D Object Detection. 2019 International Conference on Robotics and Automation

(ICRA), 2019, pp. 7276–7282, doi: 10.1109/ICRA.2019.8794195.

[16] 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 Transactions on Intelligent Transportation Systems, Vol. 23, 2022, No. 10, pp. 18040–18050, doi: 10.1109/TITS.2022.3154537.

[17] Liu, Z.—Tang, H.—Amini, A.—Yang, X.—Mao, H.—Rus, D. L.—Han, S.: BEVFusion: Multi-Task Multi-Sensor Fusion with Unified Bird's-Eye View Representation. 2023 IEEE International Conference on Robotics and Automation (ICRA), 2023, pp. 2774–2781, doi: 10.1109/ICRA48891.2023.10160968.

[18] 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. Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, No. 7, pp. 12460–12467, doi: 10.1609/aaai.v34i07.6933.

[19] Liu, Z.—Wu, Z.—Tóth, R.: SMOKE: Single-Stage Monocular 3D Object Detection via Keypoint Estimation. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2020, pp. 4289–4298, doi: 10.1109/CVPRW50498.2020.00506.

[20] Dong, L.—Liu, D.—Dong, Y.—Park, B.—Wan, Z.: An Efficient Ground Segmentation Approach for LiDAR Point Cloud Utilizing Adjacent Grids. Machine Vision and Applications, Vol. 35, 2024, No. 5, Art. No. 108, doi: 10.1007/s00138-024-01593-5.

[21] Roddick, T.—Kendall, A.—Cipolla, R.: Orthographic Feature Transform for Monocular 3D Object Detection. CoRR, 2018, doi: 10.48550/arXiv.1811.08188.

[22] Wang, Y.—Chao, W. L.—Garg, D.—Hariharan, B.—Campbell, M.—Weinberger, K. Q.: Pseudo-LiDAR from Visual Depth Estimation: Bridging the Gap in 3D Object Detection for Autonomous Driving. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 8437–8445, doi: 10.1109/CVPR.2019.00864.

[23] Li, P.—Chen, X.—Shen, S.: Stereo R-CNN Based 3D Object Detection for Autonomous Driving. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 7636–7644, doi: 10.1109/CVPR.2019.00783.

[24] Shi, S.—Wang, X.—Li, H.: PointRCNN: 3D Object Proposal Generation and Detection from Point Cloud. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 770–779, doi: 10.1109/CVPR.2019.00086.

[25] Zhang, Y.—Hu, Q.—Xu, G.—Ma, Y.—Wan, J.—Guo, Y.: Not All Points Are Equal: Learning Highly Efficient Point-Based Detectors for 3D LiDAR Point Clouds. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 18931–18940, doi: 10.1109/CVPR52688.2022.01838.

[26] Li, B.—Yan, J.—Wu, W.—Zhu, Z.—Hu, X.: High Performance Visual Tracking with Siamese Region Proposal Network. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 8971–8980, doi: 10.1109/CVPR.2018.00935.

[27] Lang, A. H.—Vora, S.—Caesar, H.—Zhou, L.—Yang, J.—Beijbom, O.: PointPillars: Fast Encoders for Object Detection from Point Clouds. 2019

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 12689–12697, doi: 10.1109/CVPR.2019.01298.

[28] O, H.—Yang, C.—Huh, K.: SeSame: Simple, Easy 3D Object Detection with Point-Wise Semantics. CoRR, 2024, doi: 10.48550/arXiv.2403.06501.

[29] Cao, P.—Chen, H.—Zhang, Y.—Wang, G.: Multi-View Frustum Pointnet for Object Detection in Autonomous Driving. 2019 IEEE International Conference on Image Processing (ICIP), 2019, pp. 3896–3899, doi: 10.1109/ICIP.2019.8803572.

[30] Wang, Z.—Jia, K.: Frustum ConvNet: Sliding Frustums to Aggregate Local Point-Wise Features for Amodal 3D Object Detection. 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019, pp. 1742–1749, doi: 10.1109/IROS40897.2019.8968513.

[31] Vora, S.—Lang, A. H.—Helou, B.—Beijbom, O.: PointPainting: Sequential Fusion for 3D Object Detection. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 4603–4611, doi: 10.1109/CVPR42600.2020.00466.

[32] Liu, W.—Lu, Y.—Zhang, T.: MFFAE-Net: Semantic Segmentation of Point Clouds Using Multi-Scale Feature Fusion and Attention Enhancement Networks. Machine Vision and Applications, Vol. 35, 2024, No. 5, Art. No. 111, doi: 10.1007/s00138-024-01589-1.

[33] Liang, M.—Yang, B.—Wang, S.—Urtasun, R.: Deep Continuous Fusion for Multi-Sensor 3D Object Detection. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (Eds.): Computer Vision – ECCV 2018. Springer, Cham, Lecture Notes in Computer Science, Vol. 11220, 2018, pp. 663–678, doi: 10.1007/978-3-030-01270-0_39.

[34] Liang, M.—Yang, B.—Chen, Y.—Hu, R.—Urtasun, R.: Multi-Task Multi-Sensor Fusion for 3D Object Detection. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 7337–7345, doi: 10.1109/CVPR.2019.00752.

[35] Pang, S.—Morris, D.—Radha, H.: Fast-CLOCs: Fast Camera-LiDAR Object Candidates Fusion for 3D Object Detection. 2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2022, pp. 3747–3756, doi: 10.1109/WACV51458.2022.00380.

[36] Guo, X.—Shi, S.—Wang, X.—Li, H.: LIGA-Stereo: Learning LiDAR Geometry Aware Representations for Stereo-Based 3D Detector. 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 3133–3143, doi: 10.1109/ICCV48922.2021.00314.

[37] Chen, Y.—Liu, S.—Shen, X.—Jia, J.: DSGN: Deep Stereo Geometry Network for 3D Object Detection. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 12533–12542, doi: 10.1109/CVPR42600.2020.01255.

[38] Lin, T. Y.—Dollár, P.—Girshick, R.—He, K.—Hariharan, B.—Belongie, S.: Feature Pyramid Networks for Object Detection. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 936–944, doi: 10.1109/CVPR.2017.106.

[39] Theckedath, D.—Sedamkar, R. R.: Detecting Affect States Using VGG16, ResNet50 and SE-ResNet50 Networks. SN Computer Science, Vol. 1, 2020, No. 2,

Art. No. 79, doi: 10.1007/s42979-020-0114-9.

[40] Zhang, K.—Hao, M.—Wang, J.—de Silva, C. W.—Fu, C.: Linked Dynamic Graph CNN: Learning on Point Cloud via Linking Hierarchical Features. CoRR, 2019, doi: 10.48550/arXiv.1904.10014.

[41] Deng, J.—Zhou, W.—Zhang, Y.—Li, H.: From Multi-View to Hollow-3D: Hallucinated Hollow-3D R-CNN for 3D Object Detection. IEEE Transactions on Circuits and Systems for Video Technology, Vol. 31, 2021, No. 12, pp. 4722–4734, doi: 10.1109/TCSVT.2021.3100848.

[42] Geiger, A.—Lenz, P.—Stiller, C.—Urtasun, R.: Vision Meets Robotics: The KITTI Dataset. The International Journal of Robotics Research, Vol. 32, 2013, No. 11, pp. 1231–1237, doi: 10.1177/0278364913491297.

[43] 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. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 11618–11628, doi: 10.1109/CVPR42600.2020.01164.

[44] Liu, Z.—Ye, X.—Tan, X.—Ding, E.—Bai, X.: StereoDistill: Pick the Cream from LiDAR for Distilling Stereo-Based 3D Object Detection. CoRR, 2023, doi: 10.48550/arXiv.2301.01615.

[45] Meng, Q.—Wang, W.—Zhou, T.—Shen, J.—Van Gool, L.—Dai, D.: Weakly Supervised 3D Object Detection from Lidar Point Cloud. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J. M. (Eds.): Computer Vision – ECCV 2020. Springer, Cham, Lecture Notes in Computer Science, Vol. 12358, 2020, pp. 515–531, doi: 10.1007/978-3-030-58601-0_31.

[46] Chen, Y.—Huang, S.—Liu, S.—Yu, B.—Jia, J.: DSGN++: Exploiting Visual-Spatial Relation for Stereo-Based 3D Detectors. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 45, 2023, No. 4, pp. 4416–4429, doi: 10.1109/TPAMI.2022.3197236.

[47] You, Y.—Wang, Y.—Chao, W. L.—Garg, D.—Pleiss, G.—Hariharan, B.—Campbell, M.—Weinberger, K. Q.: Pseudo-LiDAR++: Accurate Depth for 3D Object Detection in Autonomous Driving. CoRR, 2019, doi: 10.48550/arXiv.1906.06310.

[48] Chen, S.—Zhang, H.—Zheng, N.: Leveraging Anchor-Based LiDAR 3D Object Detection via Point Assisted Sample Selection. IEEE Transactions on Intelligent Transportation Systems, Vol. 26, 2025, No. 6, pp. 7939–7952, doi: 10.1109/TITS.2025.3555229.

[49] Gao, A.—Pang, Y.—Nie, J.—Shao, Z.—Cao, J.—Guo, Y.—Li, X.: ESGN: Efficient Stereo Geometry Network for Fast 3D Object Detection. IEEE Transactions on Circuits and Systems for Video Technology, Vol. 34, 2024, No. 4, pp. 2000–2009, doi: 10.1109/TCSVT.2022.3202810.

[50] Shi, S.—Wang, Z.—Shi, J.—Wang, X.—Li, H.: From Points to Parts: 3D Object Detection from Point Cloud with Part-Aware and Part-Aggregation Network. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 43, 2020, No. 8, pp. 2647–2664, doi: 10.1109/TPAMI.2020.2977026.

[51] Liu, Z.—Zhao, X.—Huang, T.—Hu, R.—Zhou, Y.—Bai, X.: TANet: Robust 3D Object Detection from Point Clouds with Triple Attention. Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, No. 7, pp. 11677–11684, doi: 10.1609/aaai.v34i07.6837.

[52] He, Q.—Wang, Z.—Zeng, H.—Zeng, Y.—Liu, Y.: SVGA-Net: Sparse Voxel-Graph Attention Network for 3D Object Detection from Point Clouds. Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 36, 2022, No. 1, pp. 870–878, doi: 10.1609/aaai.v36i1.19969.

[53] Liang, Z.—Zhang, Z.—Zhang, M.—Zhao, X.—Pu, S.: RangeIoUDet: Range Image Based Real-Time 3D Object Detector Optimized by Intersection over Union. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 7136–7145, doi: 10.1109/CVPR46437.2021.00706.

[54] Li, X.—Shi, B.—Hou, Y.—Wu, X.—Ma, T.—Li, Y.—He, L.: Homogeneous Multi-Modal Feature Fusion and Interaction for 3D Object Detection. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G. M., Hassner, T. (Eds.): Computer Vision – ECCV 2022. Springer, Cham, Lecture Notes in Computer Science, Vol. 13698, 2022, pp. 691–707, doi: 10.1007/978-3-031-19839-7_40.

[55] Bai, X.—Hu, Z.—Zhu, X.—Huang, Q.—Chen, Y.—Fu, H.—Tai, C. L.: TransFusion: Robust LiDAR-Camera Fusion for 3D Object Detection with Transformers. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 1080–1089, doi: 10.1109/CVPR52688.2022.00116.

[56] Chen, Y.—Li, Y.—Zhang, X.—Sun, J.—Jia, J.: Focal Sparse Convolutional Networks for 3D Object Detection. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 5418–5427, doi: 10.1109/CVPR52688.2022.00535.

[57] Chen, X.—Zhang, T.—Wang, Y.—Wang, Y.—Zhao, H.: FUTR3D: A Unified Sensor Fusion Framework for 3D Detection. 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2023, pp. 172–181, doi: 10.1109/CVPRW59228.2023.00022.

[58] Li, Y.—Chen, Y.—Qi, X.—Li, Z.—Sun, J.—Jia, J.: Unifying Voxel-Based Representation with Transformer for 3D Object Detection. In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (Eds.): Advances in Neural Information Processing Systems 35 (NeurIPS 2022). Curran Associates, Inc., 2022, pp. 18442–18455, https://proceedings.neurips.cc/paper_files/paper/2022/file/752df938681b2cf15e5fc9689f0bcf3a-Paper-Conference.pdf.

Yan Zhang received her Master's degree in mechanical electronics engineering from the China Agricultural University, Beijing China, in 2016. She began to teach at the Henan Industry and Trade Vocational College in 2023. Currently, she is Lecturer of industrial internet at the Henan Industry and Trade Vocational College. Her research interests include artificial intelligence, internet of things application technology, object detection.

Ruichao Xiao received his Master's degree in computer application technology from the China Agricultural University, Beijing China, in 2019. He began to teach at the Henan Industry and Trade Vocational College in 2020. Currently, he is a teaching assistant of computer application technology at the Henan Industry and Trade Vocational College. His research interests include computer application technology, artificial intelligence, object detection.

Lin Zhao received his Master's degree from the School of Machinery and Automation, Northeastern University. He is currently a technician at Production Management Department, China Tobacco Henan Industrial Co., Ltd., and his job title is Engineer. His research interests include computer vision and automatic control.

Xubing Li obtained his Master's degree in software engineering from the Anhui University of Science and Technology in 2024. His research interests include computer application technology, information security, system development, and artificial intelligence.

Yuankun Du received his Master's degree in computer science and technology from the Nanjing University of Science and Technology, Nanjing China, in 2013. He began to teach at the Zhengzhou University of Science and Technology in 2006. Currently, he is Associate Professor of big data and artificial intelligence at the Zhengzhou University of Science and Technology. His research interests include computer application technology, information security, system development and artificial intelligence.

Wei Wang received her B.Sc. degree in computer science and technology from Chongqing Communication Institute, Chongqing China, in 2006. She began to teach at the Henan Industry and Trade Vocational College in 2006. Currently, she is Associate Professor of computer application technology at the Henan Industry and Trade Vocational College. Her research interests include data mining, machine learning and deep learning.

Zixuan Zhang is currently studying in the School of Computer Science and Engineering, Anhui University of Science and Technology. Her research interests include computer vision, machine learning.