DH-Fusion: Depth-Aware Hybrid Feature Fusion for Multimodal 3D Object Detection

Anonymous Author(s)
Affiliation
Address
email

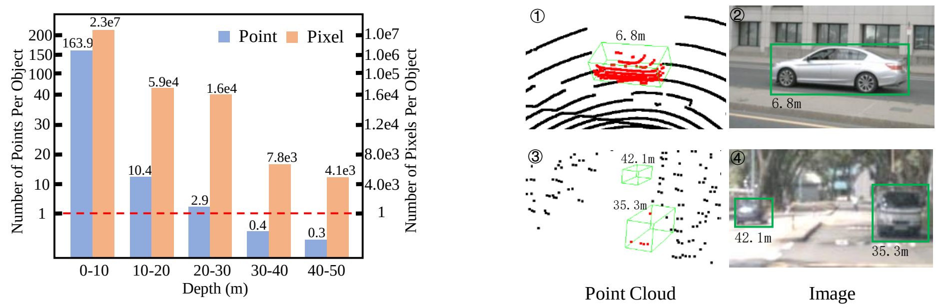
Abstract

State-of-the-art LiDAR-camera 3D object detectors usually focus on feature fusion. However, they neglect the factor of depth while designing the fusion strategy. In this work, we for the first time point out that different modalities play different roles as depth varies via statistical analysis and visualization. Based on this finding, we propose a Depth-Aware Hybrid Feature Fusion (DH-Fusion) strategy that guides the weights of point cloud and RGB image modalities by introducing depth encoding at both global and local levels. Specifically, the Depth-Aware Global Feature Fusion (DGF) module adaptively adjusts the weights of image Bird's-Eye-View (BEV) features in multi-modal global features via depth encoding. Furthermore, to compensate for the information lost when transferring raw features to the BEV space, we propose a Depth-Aware Local Feature Fusion (DLF) module, which adaptively adjusts the weights of original voxel features and multi-view image features in multi-modal local features via depth encoding. Extensive experiments on the nuScenes dataset demonstrate that our DH-Fusion method surpasses previous state-of-the-art methods w.r.t. NDS. Moreover, our DH-Fusion is more robust to various kinds of corruptions, outperforming previous methods on nuScenes-C w.r.t. both NDS and mAP.

1 Introduction

3D object detection has a wide range of applications in the fields of autonomous driving and robotics. A large number of previous works have successfully focused on using a single modality, such as point cloud or images, to design efficient 3D object detectors. However, the performance of these detectors reaches a bottleneck due to the limitations of modality characteristics. For instance, the point cloud modality can only provide rich geometric information while lacks detailed semantic information; the image modality can only provide rich texture information while lacks three-dimensional spatial information. To address the aforementioned issues, we are highly motivated to obtain comprehensive information that represents objects by designing a LiDAR-camera 3D object detector.

In recent years, LiDAR-camera 3D object detection develops rapidly. Some works [1, 4, 28, 33, 67] propose effective methods to integrate information from two modalities at the feature level. However, they all overlook an important factor of depth in their fusion strategies. To understand how point cloud and image information vary with depth, we first conduct statistical and visualization analysis on the nuScenes-mini dataset [3], and find that: (1) The number of points representing objects at near range is relatively large, which allows us to accurately determine the object's location, size, and category, even without the aid of images. As shown in Fig. 1a, there is an average of 163.7 points per object within 0-10 meters, which is a substantial number. We also visualize a car at 6.8 meters in Fig. 1b ① and find it encompasses a considerable number of points, well representing the shape. In contrast, some background noise in the image may interfere with detection (Fig. 1b ②). (2) As the



(a) Statistical chart (b) Visualization

Figure 1: Statistical and visualization analysis on the nuScenes-mini dataset. (a) The average numbers of points and pixels for each object at different depths. (b) Examples of near-range and long-range objects in images and point cloud. Points within the bounding boxes are colored red for observation.

depth increases, the number of points representing objects decreases rapidly. As shown in Fig. 1a, the number of points within 30-50 meters falls below one per object, meaning that many objects are even not represented by any points, such as the object at 42.1 meters in Fig. 1b ③. In contrast, the complete objects may still be observed on the image, as in Fig. 1b ④, where the image information becomes more important. To address the above problems, we propose a feature fusion strategy that adaptively adjusts the importance of the two modalities based on depth.

Specifically, we propose a novel method for multi-modal 3D object detection, namely Depth-Aware Hybrid Feature Fusion (DH-Fusion). The innovation lies in adaptively adjusting the weights of features by introducing depth encoding to hybrid feature fusion at both global and local levels. The fusion strategy consists of two crucial components: Depth-Aware Global Feature Fusion (DGF) module and Depth-Aware Local Feature Fusion (DLF) module. In DGF, we take point cloud Bird's-Eye-View (BEV) features and image BEV features as inputs, and dynamically adjust the weights of image BEV features based on depth during fusion by utilizing a global-fusion transformer encoder with a depth encoder. To compensate for the information lost when transforming raw features to BEV space, we enhance the fused BEV features at a lower cost by utilizing the original instance features. In DLF, we obtain 3D boxes by utilizing a Region Proposal Network (RPN). Then, the 3D boxes are projected into both LiDAR voxel features and multi-view image features to crop out corresponding local instance features with more detailed information. Afterward, we take these as inputs and dynamically adjust the weights of local multi-view image features and local LiDAR voxel features based on depth through the use of a local-fusion transformer encoder with the depth encoder. In the end, we update local features for each object on the global feature map to enhance the detailed instance information of multi-modal global features for detection.

Our contributions are summarized as follows.

1. We for the first time point out that depth is an important factor to consider while fusing LiDAR point cloud features and RGB image features for 3D object detection. From our statistical and visualization analysis, we can see that image features play different roles as depth varies.

2. We propose a depth-aware hybrid feature fusion strategy that dynamically adjusts the weights of features during feature fusion by introducing depth encoding at both global and local levels. The above strategy can obtain high-quality features for detection, fully leveraging the advantages of different modalities at various depths.

3. Our method is evaluated on the nuScenes [3] dataset and a more challenging nuScenes-C [13] dataset, outperforming previous multi-modal methods and being robust to various kinds of data corruptions.

2 Related Work

Since our method is based on conducting 3D object detection using data from multiple modalities, including point cloud and images, we briefly review recent works in the following fields: LiDAR-based 3D object detection, camera-based 3D object detection, and LiDAR-camera 3D object detection.

2.1 LiDAR-based 3D Object Detection

LiDAR-based 3D object detectors only take the point cloud as input. Based on their different data representations, they can be divided into point-based [44–46, 64, 65], voxel-based [12, 22, 61, 68, 71], and point-voxel-based [17, 42, 43] methods. The feature extraction networks of point-based methods typically extract features directly from the point cloud through a point-based backbone [40], such as PointRCNN [44]. The voxel-based methods first convert the point cloud into voxels and then extract voxel features through a 3D sparse convolution network [14], such as VoxelNet [71]. Point-voxel-based methods like PV-RCNN [42] combine the above two methods to extract and fuse point and voxel features. The purpose of these approaches is to capture the geometric spatial information of the point cloud. However, point cloud is sparse and incomplete, lacking detailed texture information, which greatly limits the detection performance.

2.2 Camera-based 3D Object Detection

Camera-based 3D object detectors only take images as inputs. Depending on the form of inputs, they can be divided into monocular [2, 24, 32, 41, 47, 55], stereo [6, 25, 30, 48, 70], and multi-view [19, 27, 56, 62] 3D object detectors. Early works like FCOS3D [55] input a monocular image and utilize 2D object detectors to directly predict 3D bounding boxes, but these approaches have limited capability in capturing spatial information. Subsequently, stereo and multi-view 3D object detectors are proposed to obtain more precise depth information by constructing spatial relationships among multiple images, such as Stereo RCNN [25] and BEVDet [19]. These methods successfully achieve purely visual 3D object detection, but they do not perform as well as LiDAR-based methods, because the spatial depth information provided by images is not as direct and precise as that provided by point cloud.

2.3 LiDAR-Camera 3D Object Detection

LiDAR-camera 3D object detectors take point cloud and images as inputs, and can be classified into early-fusion-based [50, 52, 57, 59, 69], intermediate-fusion-based [1, 4, 28, 33, 67], and late-fusion-based [37, 38] 3D object detectors based on the location of multi-modal information fusion [36].

Early-fusion-based methods perform at the point level, where the typical approach involves enhancing the raw point cloud with semantic information extracted from images. PointPainting [50] and FusionPainting [59] decorate the raw point cloud with semantic scores from 2D semantic segmentation. Similarly, PointAugmenting [52] enhances the raw point cloud using features extracted from a 2D semantic segmentation network. However, early-fusion-based methods are sensitive to alignment errors between the two modalities.

Intermediate-fusion-based methods perform at the feature level. Transfusion [1] first proposes to utilize the transformer for fine-grained fusion from LiDAR BEV features and multi-view image features. FUTR3D [5] encode each modality using deformable attention [73] in its own coordinate and concatenate them for fusion. BEVFusion [28, 33] projects both point cloud and images to BEV space for BEV feature fusion. SparseFusion [58] extracts instance-level features from both two modalities separately, and fuse them to perform detection. Similarly, ObjectFusion [4] utilizes 3D proposals from LiDAR modality to extract instance-level features for fusion. CMT [60] proposes the simultaneous interaction between the object queries and multi-modal features in the transformer encoder and decoder. IS-Fusion [67] proposes feature fusion at both the instance level and scene level. The intermediate-fusion-based methods gradually become a mainstream approach due to the diversity of fusion strategies.

Late-fusion-based methods perform at the bounding box level. Typically, CLOCs [37] obtains 2D and 3D bounding boxes by separately using 2D and 3D object detectors, and then combine them to achieve more accurate 3D bounding boxes. However, the interaction between modalities in late-fusion-based methods is very limited, which constrains model performance.

These multi-modal methods successfully outperform single-modal methods. However, their feature fusion methods do not take depth into account. In contrast, our approach introduces depth information to guide the hybrid feature fusion, boosting the performance of the detector.

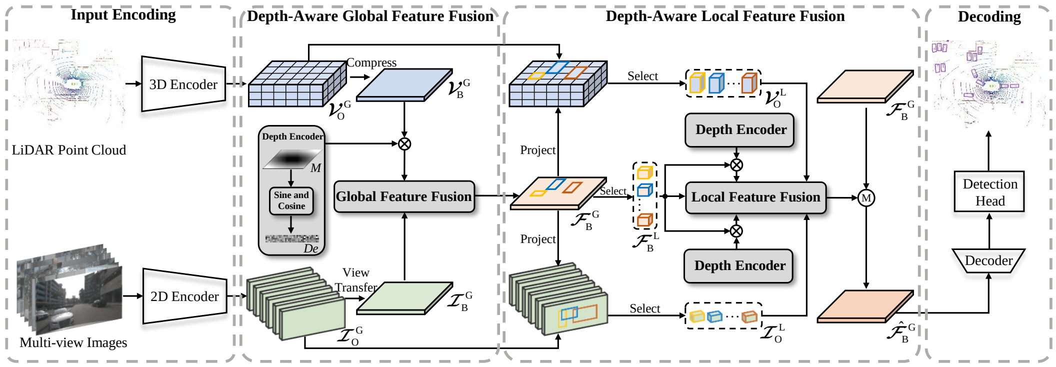


Figure 2: Overview of our method. It introduces depth encoding in both global and local feature fusion to obtain depth-adaptive multi-modal representations for detection. $\otimes$ is the multiplication operation, and Ⓜ is the merge operation.

3 Methodology

In this section, we first give an overview of our proposed multi-modal 3D object detector, and then provide a detailed introduction to our proposed feature fusion method.

3.1 Overview

We propose a multi-modal 3D object detection method via Depth-Aware Hybrid Feature Fusion (DH-Fusion). As illustrated in Fig. 2, our approach consists of two important feature fusion modules: Depth-Aware Global Feature Fusion (DGF) and Depth-Aware Local Feature Fusion (DLF). In the following, we briefly describe the detection pipeline.

Inputs. First, we take the point cloud P and multi-view images I as inputs, where point cloud consists of a set of points: $P = \{P_1, P_2, \cdots, P_{N_l}\}$, and each point has four dimensions: X-axis, Y-axis, Z-axis, and intensity; the multi-view images comprise N_c images: $I = \{I_1, I_2, \cdots, I_{N_c}\}$, each image captured by its corresponding camera.

Input Encoding. For the point cloud P, we use a 3D encoder to extract raw global voxel features $\mathcal{V}_O^G$; for the multi-view images I, we use a 2D encoder to extract image features of all views $\mathcal{I}_O^G$.

Hybrid Feature Fusion. Then, for voxel features $\mathcal{V}_O^G$, we compress the height dimension to obtain point cloud BEV features $\mathcal{V}_B^G$; for image features $\mathcal{I}_O^G$, we transform their perspective view to bird's eye view to obtain image BEV features $\mathcal{I}_B^G$. To fully leverage the features from two modalities, we design a DGF module that aims to dynamically adjust the weights of image BEV features based on depth values during feature fusion. Please refer to Sec. 3.2 for more details. To compensate for the information lost when transforming raw features to BEV space, we propose a DLF module that, based on depth, utilizes the raw features to enhance the detailed information of each object instance in global multi-modal features. It consists of three processes: local feature selection, local feature fusion, and merging local features into global features. First, we obtain the local multi-modal BEV features $\mathcal{F}_B^L$, local voxel features $\mathcal{V}_O^L$, and local multi-view image features $\mathcal{I}_O^L$, by cropping the corresponding global features based on the 3D boxes obtained from an RPN; then, it dynamically and individually adjusts the weights of each local feature of $\mathcal{V}_O^L$ and $\mathcal{I}_O^L$ based on depth values during feature fusion; finally, we update local features for each object on the global feature map. Please refer to Sec. 3.3 for more details. In this way, we obtain enhanced multi-modal global features for detection.

Decoding. Based on the enhanced multi-modal global features $\hat{\mathcal{F}}_B^G$ that contain rich semantic and spatial information, we utilize a transformer decoder and a detection head to predict the object categories and 3D bounding boxes.

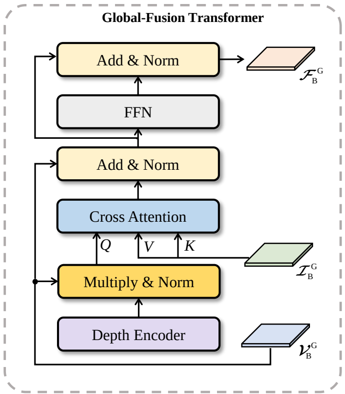


Figure 3: Illustration of the DGF. It consists of a global fusion transformer with the depth encoder.

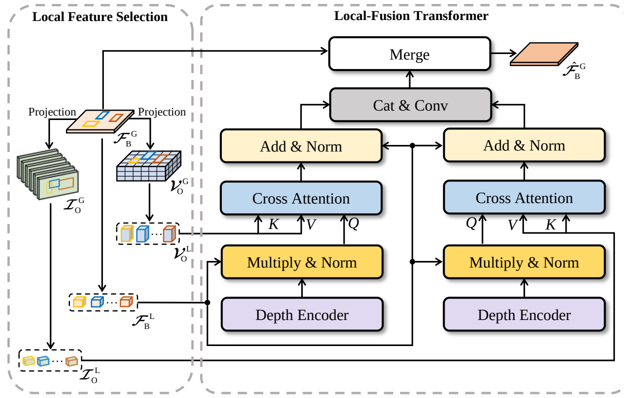


Figure 4: Illustration of the DLF. It consists of a local feature selection module and a local fusion transformer with the depth encoder.

3.2 Depth-Aware Global Feature Fusion

As shown in Fig. 3, the DGF module consists of a global-fusion transformer with a depth encoder. In the following, we provide a detailed explanation of each component.

3.2.1 Depth Encoder

We introduce depth encoding (DE) in feature fusion to dynamically adjust the weights of image BEV features during fusion. First, we build a depth matrix M to store the depth value of each position element p_k represented as:

$$p_k = \{(x_k, y_k) : d_k\}, k \in [1, n], \tag{1}$$

where (x_k, y_k) are the positional coordinates, d_k is the depth value, and n is the number of elements. Then, we use Euclidean distance to calculate the distance between every element's spatial location (x_k, y_k) and the ego coordinate element's location $(x_{\frac{n}{2}}, y_{\frac{n}{2}})$:

$$d_k = E((x_k, y_k), (x_{\frac{n}{2}}, y_{\frac{n}{2}})), k \in [1, n], \tag{2}$$

where we denote $E(\cdot)$ as the Euclidean distance calculation. The depth matrix M serves as a lookup table to avoid redundant computation of depth values. Since the size of the BEV features is large and the depth distribution is simple, to avoid introducing additional parameters, the depth encoding De is obtained by applying sine and cosine functions [49] to the depth matrix.

3.2.2 Global-Fusion Transformer

In the global-fusion transformer, we take the point cloud BEV features $\mathcal{V}_B^G \in \mathbb{R}^{W \times H \times C}$ and image BEV features $\mathcal{I}_B^G \in \mathbb{R}^{W \times H \times C}$ as inputs, and integrate the depth encoding obtained above by multiplying it with the point cloud BEV features, forming the query $Q_{\mathcal{V}}^G = N(\mathcal{V}_B^G \times Conv(De))$, where $Conv(\cdot)$ is a convolution operation to align with the channels of $\mathcal{V}_B^G$, and $N(\cdot)$ is a normalization layer. The image BEV features are queried as the corresponding key $K_{\mathcal{I}}^G$ and value $V_{\mathcal{I}}^G$. We utilize the multi-head cross attention to achieve the interacted feature $\hat{\mathcal{V}}_B^G$ based on depth:

$$\hat{\mathcal{V}}_B^G = CA(Q_{\mathcal{V}}^G, K_{\mathcal{I}}^G, V_{\mathcal{I}}^G), \tag{3}$$

where $CA(\cdot)$ indicates the multi-head cross attention. Afterward, we aggregate the information from both modalities to obtain the fused features $\mathcal{F}_B^G$:

$$\mathcal{F}_B^G = N(FFN(N(\hat{\mathcal{V}}_B^G + \mathcal{V}_B^G)) + N(\hat{\mathcal{V}}_B^G + \mathcal{V}_B^G)), \tag{4}$$

where $N(\cdot)$ is a normalization layer; $FFN(\cdot)$ specifies a feed-forward network containing two convolution operations. In this way, we obtain fused features in which the image features play different roles as the depth varies.

3.3 Depth-Aware Local Feature Fusion

As shown in Fig. 4, the DLF module consists of a local feature selection and a local-fusion transformer with the depth encoder. In the following, we provide a detailed explanation of each component.

3.3.1 Local Feature Selection

To compensate for the information lost when transforming point cloud features and image features to BEV space, we enhance the instance details of fused BEV features $\mathcal{F}_B^G$ using instance features from raw voxel features $\mathcal{V}_O^G$ and multi-view image features $\mathcal{I}_O^G$. Specifically, we utilize an RPN to regress t 3D boxes based on the BEV features $\mathcal{F}_B^G$. We directly crop the global fused BEV features $\mathcal{F}_B^G$ based on the regressed 3D boxes to obtain the local fused BEV features $\mathcal{F}_B^L \in \mathbb{R}^{c\times t}$. On the other hand, we project the 3D boxes onto the raw voxel features and multi-view image features to obtain their corresponding local features before global fusion, preserving richer information for each object instance. Specifically, we utilize the voxel pooling operation [12], followed by a 3D convolution operation and a linear layer, to extract local voxel features $\mathcal{V}_O^L \in \mathbb{R}^{c\times t}$; we transform the 3D boxes from bird's eye view to perspective view, and utilize the RoI Align operation [15], followed by a linear layer, to extract instance image features $\mathcal{I}_O^L \in \mathbb{R}^{c\times t}$. By doing this, we obtain the hybrid (before & after global fusion) local features, which will be sent to the subsequent fusion module.

3.3.2 Local-Fusion Transformer

In the local-fusion transformer, the weights of each local raw feature are dynamically adjusted based on depth values during feature fusion, and we update local features for each object on the global feature map. Specifically, we take the local multi-modal BEV features $\mathcal{F}_B^L$, local voxel features $\mathcal{V}_O^L$, and local multi-view image features $\mathcal{I}_O^L$ as inputs, and integrate the depth encoding by multiplying it with the local multi-modal BEV features, forming the query $Q_\mathcal{F}^L$. The local multi-view image features and local voxel features are respectively queried as the corresponding key $K_\mathcal{I}^L$, $K_\mathcal{V}^L$ and value $V_\mathcal{I}^L$, $V_\mathcal{V}^L$. The two multi-head cross-attention modules are utilized to achieve the interacted features $\hat{Q}_\mathcal{F}^L, \hat{Q}_\mathcal{F}^{L\prime}$. Note that the computation process of multi-head cross attention is similar to that described in Sec. 3.2.2 and is omitted here. Afterward, we aggregate the above features:

$$\hat{\mathcal{F}}_B^L = Conv(Cat(\hat{Q}_\mathcal{F}^L + \mathcal{F}_B^L, \hat{Q}_\mathcal{F}^{L\prime} + \mathcal{F}_B^{L\prime})), \tag{5}$$

where $Cat(\cdot)$ is the concatenation operation; $Conv(\cdot)$ is used to align with the feature channels of global fused BEV features $\mathcal{F}_B^G$. As a result, we obtain enhanced local features by dynamically calling back rich information in raw modalities at various depths. Afterward, we update the global features $\mathcal{F}_B^G$ by inserting the enhanced local features at corresponding locations.

4 Experiments

In this section, we will first introduce the dataset and evaluation metrics, followed by the implementation details. Then, we will compare our method with the state-of-the-art methods on nuScenes and also present results on a more challenging dataset of nuScenes-C with data corruptions. Finally, we will show the ablation studies and qualitative results. More experiments are provided in Appendix A.2.

4.1 Experimental Setup

Datasets and evaluation metrics. We evaluate our proposed DH-Fusion on the nuScenes benchmark [3] and a more challenging dataset of nuScenes-C [13] with data corruptions. nuScenes dataset provides 700 scene sequences for training, 150 scene sequences for validation, and 150 scene sequences for testing. Each sequence contains 40 frames of 32-beam LiDAR data, and each frame has six corresponding images covering a 360-degree field of view. It offers calibration matrices that facilitate accurate projection of 3D points onto 2D pixels, and contains 10 object categories that are commonly encountered within autonomous driving. nuScenes-C dataset provides 27 corruptions with 5 severities on the nuScenes validation set, including corruptions at the weather, sensor, motion, object, and alignment level. We use the nuScenes detection scores (NDS) and mean Average Precision (mAP) to evaluate our detection results, where NDS is a comprehensive metric in nuScenes that combines object translation, scale, orientation, velocity, and attribute errors.

Implementation details. We implement the proposed DH-Fusion with PyTorch [39] under the open-source framework MMDetection3D [10]. Specifically, for the LiDAR branch, we use VoxelNet [71] with FPN [61] as the 3D encoder. The voxel size is set to [0.075m, 0.075m, 0.1m], and the range of point cloud is [-54m, 54m] along the X-axis, [-54m, 54m] along the Y-axis, and [-3m, 5m] along the Z-axis. For the image branch, we use the ResNet18 [16], ResNet50 [16], and SwinTiny [34] with FPN [29] as the 2D image encoder of DH-Fusion-light, -base, -large, respectively. Correspondingly, the resolution of input images is resized to 256×704, 320×800, and 384×1056. Additionally, we utilize BEVPoolV2 [18] to obtain image BEV features. Following [33], the feature size $W \times H$ is set to 180×180, the channel C is set to 128, and the channel c is also set to 128. The multi-head cross attention is implemented with 8 heads, and the FFN contains 2 MLP layers with a hidden dimension of 128. Following [58], the number of regressed 3D boxes t is set to 200. More implementation details are provided in Appendix A.1.

4.2 Comparison to the State of the Art

Aiming for a fair comparison, we categorize previous methods based on the types of 2D backbones into ResNet50-based, SwinTiny-based, and others, and provide three versions of our proposed method, named DH-Fusion-light, DH-Fusion-base, and DH-Fusion-large. The results are shown in Tab. 1. (1) Compared with the ResNet50-based methods, our DH-Fusion-base outperforms the top method FocalFormer3D [7] by up to 1 pp w.r.t. NDS under the same configuration. Specifically, we reach 74.0% w.r.t. NDS and 71.2% w.r.t. mAP on the validation set, and 74.7% w.r.t. NDS and 71.7% w.r.t. mAP on the test set, while maintaining comparable inference speed of 8.7 FPS on a 3090 GPU. (2) Compared with the SwinTiny-based methods and others, our DH-Fusion-large outperforms the top method IS-Fusion [67] under the same configuration, and runs 2x faster than it. Specifically, we reach 74.4% w.r.t. NDS on the validation set, and 75.4% w.r.t. NDS on the test set, while achieving a faster inference speed of 5.7 FPS on a 3090 GPU, indicating that our proposed method is both more effective and efficient. (3) Furthermore, our DH-Fusion-light surpasses the typical BEVFusion [33] by up to 1 pp w.r.t. all metrics using a lighter 2D backbone, and achieves a real-time inference speed of 13.8 FPS. Overall, our method achieves higher detection accuracy and faster inference speed.

4.3 Robustness to Corruptions

We further implement some experiments on the nuScenes-C [13] dataset to evaluate the model's robustness under various corruptions, including changes in weather, data loss or temporal-spatial misalignment in multi-modal inputs, etc. The results for different kinds of corruptions are shown in Tab. 2, and more detailed results for each fine-grained corruption are shown in Appendix A.2.3. We find that our DH-Fusion-light still achieves an average performance of 68.67% w.r.t. NDS and 63.07% w.r.t. mAP under various corruptions, which only decreases by 4.63 pp w.r.t. NDS and 6.68 pp w.r.t. mAP, compared to its performance without corruptions. Performance drop is smaller than that observed with previous methods including BEVFusion [28] across all kinds of corruptions, indicating that our DH-Fusion-light possesses superior robustness. Furthermore, we observe that our DH-Fusion-light is particularly robust against weather and object corruptions, where the performance drop is less than 3pp. The more stable performance indicates that our method is more friendly to practical applications, where data corruption may occur.

4.4 Ablation Studies

We conduct ablation studies to first demonstrate the effect of each component of DH-Fusion, then to demonstrate the effect of depth encoding in DGF and DLF, and finally to assess the impact of multiplying depth encoding. All method variants are implemented on the nuScenes validation dataset.

Table 1: Comparisons with the state of the art on the nuScenes `validation` and `test` sets. FPS is measured on a 3090 GPU by default, and * denotes the inference speed on an A100 GPU referred from the original paper. Note that all results are obtained without any model ensemble or test time augmentation.

Methods	Present at	Image Size - 2D Backbone	FPS	Validation NDS	Validation mAP	Test NDS	Test mAP
Image Backbone: ResNet50[16]							
Trainsfusion [1]	CVPR'22	320 × 800-ResNet50	6.5	71.3	67.5	71.7	68.9
DeepInteraction [66]	NeurIPS'22	448 × 800-ResNet50	1.9	72.4	69.9	73.4	70.8
MSMDFusion [21]	CVPR'23	448 × 800- ResNet50	2.1	72.1	69.7	74.0	71.5
FocalFormer3D [7]	ICCV'23	320 × 800-ResNet50	9.2*	73.1	70.1	73.9	71.6
DH-Fusion-base (Ours)	-	320 × 800-ResNet50	8.7	**74.0**	**71.2**	**74.7**	**71.7**
Image Backbone: SwinTiny[31]							
BEVFusion [28]	NeurIPS'22	448 × 800-SwinTiny	0.7*	71.0	67.9	71.8	69.2
BEVFusion [33]	ICRA'23	256 × 704- SwinTiny	9.6	71.4	68.5	72.9	70.2
ObjectFusion [4]	ICCV'23	256 × 704- SwinTiny	-	72.3	69.8	73.3	71.0
SparseFusion [58]	ICCV'23	256 × 704- SwinTiny	4.4	72.8	70.5	73.8	72.0
IS-Fusion [67]	CVPR'24	384 × 1056-SwinTiny	3.2*	74.0	**72.8**	75.2	**73.0**
Image Backbone: Others							
AutoAlignV2 [8]	ECCV'22	640 × 1280-CSPNet [51]	4.8*	71.2	67.1	72.4	68.4
UVTR [26]	NeurIPS'22	640 × 1280-ResNet101 [16]	1.8	70.2	65.4	71.1	67.1
FUTR3D [5]	CVPR'23	900 × 1600-VOVNet [23]	3.3*	68.0	64.2	72.1	69.4
UniTR [54]	ICCV'23	256 × 704-DSVT [53]	9.3*	73.3	70.5	74.5	70.9
CMT [60]	ICCV'23	640 × 1600-VOVNet	6.0*	72.9	70.3	74.1	72.0
UniPAD [63]	CVPR'24	900 × 1600-ConvNeXtS [34]	-	73.2	69.9	73.9	71.0
DH-Fusion-large (Ours)	-	384 × 1056-SwinTiny	5.7	**74.4**	72.3	**75.4**	72.8
DH-Fusion-light (Ours)	-	256 × 704-ResNet18	**13.8**	73.3	69.8	74.2	70.9

Table 2: Robustness experiments on nuScenes-C. Numbers are **NDS / mAP**.

Methods	Corruption						Average
	None	Weather	Sensor	Motion	Object	Alignment	
FUTR3D [5]	68.05 / 64.17	62.75 / 55.51	63.66 / 56.83	53.16 / 44.43	65.45 / 61.04	62.83 / 57.60	$62.82^{\downarrow 5.23}$ / $56.99^{\downarrow 7.18}$
TransFusion [1]	69.82 / 66.38	65.42 / 59.37	66.17 / 59.82	51.52 / 41.47	68.28 / 64.38	61.98 / 54.94	$63.74^{\downarrow 6.08}$ / $58.73^{\downarrow 7.65}$
BEVFusion [33]	71.40 / 68.45	67.54 / 61.87	67.59 / 61.80	55.19 / 47.30	68.01 / 65.14	63.94 / 58.71	$66.06^{\downarrow 5.34}$ / $61.03^{\downarrow 7.42}$
DH-Fusion-light (Ours)	**73.30 / 69.75**	**72.19 / 67.48**	**69.16 / 62.87**	**57.07 / 47.52**	**71.01 / 67.11**	**67.24 / 62.38**	**$68.67^{\downarrow 4.63}$ / $63.07^{\downarrow 6.68}$**

Effect of DGF and DLF. To demonstrate the effect of DGF and DLF, we conduct experiments by integrating the components one by one into the baseline, BEVFusion [33]. The results are shown in Tab. 3. We find that our DGF improves the baseline performance by 1.0 pp w.r.t. NDS and 0.9 pp w.r.t. mAP. This demonstrates that dynamically adjusting the weights of the image BEV features during fusion is effective for 3D object detection. Additionally, our DLF improves the baseline performance by 1.3 pp w.r.t. NDS and 0.8 pp w.r.t. mAP, which indicates that dynamically adjusting the weights of the local raw instance features based on depth during fusion effectively compensates for the information loss caused by the transformation of global features into the BEV feature space. The results of integrating both components show an improvement of 1.9 pp w.r.t. NDS and 1.3 pp w.r.t. mAP, well verifying the benefits of dynamically fusing global and local hybrid features based on depth.

Effect of depth encoding in DGF and DLF. To evaluate the effectiveness of our depth encoding, we conduct experiments where the depth encoding is removed from the DGF and DLF modules, respectively. The results are shown in Tab. 4. When removing the depth encoding from Baseline+DGF, the performance drops by 0.6 pp w.r.t. NDS and 0.4 pp w.r.t. mAP. Similarly, when removing the depth encoding from Baseline+DLF, the performance also decreases by 1.1 pp w.r.t. NDS and 0.9 pp w.r.t. mAP. These results indicate that our depth encoding is effective. Furthermore, we observe that removing the depth encoding from the DLF module results in a larger performance drop, suggesting that depth encoding plays a more crucial role in local feature fusion.

Impact of different operations for depth encoding. We conduct experiments with different operations of depth encoding, including concatenation, summation, and multiplication. The results in Tab. 5, show that the multiplication operation consistently outperforms the summation and concatenation operations w.r.t. both metrics. The superior performance of multiplication can be attributed to its ability to more effectively modulate the feature maps based on depth information. Unlike summation, which simply shifts the feature values, or concatenation, which increases the dimensionality without direct interaction, multiplication allows for more interaction between the

Table 3: Ablation studies of each proposed module.

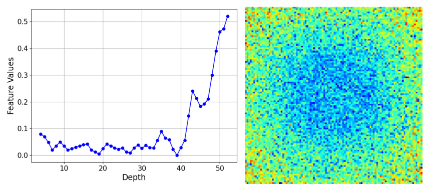
Baseline	DGF	DLF	NDS	mAP
✓			71.4	68.5
✓	✓		72.4$^{\uparrow 1.0}$	69.4$^{\uparrow 0.9}$
✓		✓	72.7$^{\uparrow 1.3}$	69.3$^{\uparrow 0.8}$
✓	✓	✓	**73.3**$^{\uparrow 1.9}$	**69.8**$^{\uparrow 1.3}$

Table 4: Ablation studies of depth encoding (DE) in DGF and DLF.

Methods	NDS	mAP
Baseline + DGF	72.4	69.4
w/o DE	71.8$^{\downarrow 0.6}$	69.0$^{\downarrow 0.4}$
Baseline + DLF	72.7	69.3
w/o DE	71.6$^{\downarrow 1.1}$	68.4$^{\downarrow 0.9}$

Table 5: Ablation studies of different operations for depth encoding.

Methods	NDS	mAP
Summation	72.8	69.2
Concatenation	72.5	68.7
Multiplication	**73.3**	**69.8**



(a) Attention weights (b) Average map

Figure 5: Attention weights applied on BEV image features in DGF vary with depth.

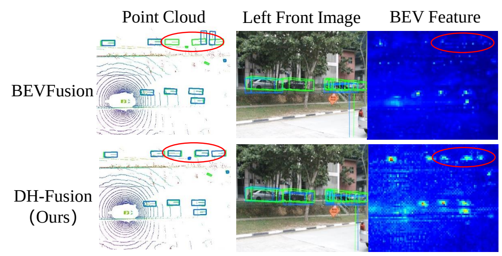


Figure 6: Qualitative detection results and BEV features of BEVFusion and ours. We show the ground truth boxes in green, and the prediction boxes in blue.

depth encoding and features, leading to better feature representation and ultimately improving the detection performance.

4.5 Qualitative Results

To better understand how depth encoding affects the feature fusion, in Fig. 5, we plot a curve to observe how the attention weights applied on the image BEV features in our DGF module vary with depth, and visualize the average attention map. It is evident that the weights of the image BEV features stay low in near range, but go up significantly as depth increases when the depth is larger than 40 meters. This trend supports our hypothesis that the image modality would become more important as depth increases. In this way, our depth encoding allows the model to dynamically adjust the weights of image BEV features based on depth.

We also compare the detection results of our DH-Fusion method with the baseline BEVFusion [33] in Fig. 6, where we clearly find that our method better localizes those distant objects compared to BEVFusion. These results demonstrate that our proposed multi-modal fusion strategy based on depth is more effective for detection. Besides, we exhibit the corresponding BEV feature maps, where our method shows a stronger feature response for the foreground objects, especially for distant ones. That is why our feature fusion strategy can provide higher-quality detection results. More qualitative results can be found in Appendix A.3.

5 Conclusion

In this paper, we for the first time point out that different modalities play different roles as depth varies via statistical analysis and visualization. Based on this finding, we propose a feature fusion strategy for multi-modal 3D object detection, namely Depth-Aware Hybrid Feature Fusion (DH-Fusion), that dynamically adjusts the weights of features during feature fusion by introducing depth encoding at both global and local levels. Extensive experiments on the nuScenes dataset demonstrate that our DH-Fusion method surpasses previous state-of-the-art methods w.r.t. NDS. Moreover, our DH-Fusion is more robust to various kinds of corruptions, outperforming previous methods on the nuScenes-C dataset w.r.t. both NDS and mAP. Our method uses an attention-based approach to interact with the two modalities, making the detection results sensitive to modality loss. We plan to further explore feature fusion methods that are robust to modality loss. Although our method improves detection performance, emergency plans still need to be implemented in practical applications to ensure personnel safety.

References

[1] 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. In: CVPR (2022)

[2] Brazil, G., Liu, X.: M3d-rpn: Monocular 3d region proposal network for object detection. In: ICCV (2019)

[3] 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: CVPR (2020)

[4] Cai, Q., Pan, Y., Yao, T., Ngo, C.W., Mei, T.: Objectfusion: Multi-modal 3d object detection with object-centric fusion. In: ICCV (2023)

[5] Chen, X., Zhang, T., Wang, Y., Wang, Y., Zhao, H.: Futr3d: A unified sensor fusion framework for 3d detection. In: CVPR (2023)

[6] Chen, Y., Liu, S., Shen, X., Jia, J.: Dsgn: Deep stereo geometry network for 3d object detection. In: CVPR (2020)

[7] Chen, Y., Yu, Z., Chen, Y., Lan, S., Anandkumar, A., Jia, J., Alvarez, J.M.: Focalformer3d: focusing on hard instance for 3d object detection. In: ICCV (2023)

[8] Chen, Z., Li, Z., Zhang, S., Fang, L., Jiang, Q., Zhao, F.: Deformable feature aggregation for dynamic multi-modal 3d object detection. In: ECCV (2022)

[9] Chiu, H.k., Prioletti, A., Li, J., Bohg, J.: Probabilistic 3d multi-object tracking for autonomous driving. arxiv 2020. arXiv preprint arXiv:2001.05673 (2020)

[10] Contributors, M.: MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. `https://github.com/open-mmlab/mmdetection3d` (2020)

[11] Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)

[12] Deng, J., Shi, S., Li, P., Zhou, W., Zhang, Y., Li, H.: Voxel r-cnn: Towards high performance voxel-based 3d object detection. In: AAAI (2021)

[13] Dong, Y., Kang, C., Zhang, J., Zhu, Z., Wang, Y., Yang, X., Su, H., Wei, X., Zhu, J.: Benchmarking robustness of 3d object detection to common corruptions. In: CVPR (2023)

[14] Graham, B., Engelcke, M., Van Der Maaten, L.: 3d semantic segmentation with submanifold sparse convolutional networks. In: CVPR (2018)

[15] He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: CVPR (2017)

[16] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)

[17] Hu, J.S., Kuai, T., Waslander, S.L.: Point density-aware voxels for lidar 3d object detection. In: CVPR (2022)

[18] Huang, J., Huang, G.: Bevpoolv2: A cutting-edge implementation of bevdet toward deployment. arXiv:2211.17111 (2022)

[19] Huang, J., Huang, G., Zhu, Z., Ye, Y., Du, D.: Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. arXiv:2112.11790 (2021)

[20] Huang, J., Ye, Y., Liang, Z., Shan, Y., Du, D.: Detecting as labeling: Rethinking lidar-camera fusion in 3d object detection. arXiv arXiv:2311.07152 (2023)

[21] Jiao, Y., Jie, Z., Chen, S., Chen, J., Ma, L., Jiang, Y.G.: Msmdfusion: Fusing lidar and camera at multiple scales with multi-depth seeds for 3d object detection. In: CVPR (2023)

[22] Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: CVPR (2019)

[23] Lee, Y., Hwang, J.w., Lee, S., Bae, Y., Park, J.: An energy and gpu-computation efficient backbone network for real-time object detection. In: CVPR workshops (2019)

[24] Li, B., Ouyang, W., Sheng, L., Zeng, X., Wang, X.: Gs3d: An efficient 3d object detection framework for autonomous driving. In: CVPR (2019)

[25] Li, P., Chen, X., Shen, S.: Stereo r-cnn based 3d object detection for autonomous driving. In: CVPR (2019)

[26] Li, Y., Chen, Y., Qi, X., Li, Z., Sun, J., Jia, J.: Unifying voxel-based representation with transformer for 3d object detection. In: NeurIPS (2022)

[27] Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Qiao, Y., Dai, J.: Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In: ECCV (2022)

[28] Liang, T., Xie, H., Yu, K., Xia, Z., Lin, Z., Wang, Y., Tang, T., Wang, B., Tang, Z.: Bevfusion: A simple and robust lidar-camera fusion framework. In: NeurIPS (2022)

[29] Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: CVPR (2017)

[30] Liu, Y., Wang, L., Liu, M.: Yolostereo3d: A step back to 2d for efficient stereo 3d detection. In: ICRA. IEEE (2021)

[31] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: ICCV (2021)

[32] Liu, Z., Wu, Z., Tóth, R.: Smoke: Single-stage monocular 3d object detection via keypoint estimation. In: CVPR (2020)

[33] 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. In: ICRA (2023)

[34] Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. In: CVPR (2022)

[35] Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)

[36] Mao, J., Shi, S., Wang, X., Li, H.: 3d object detection for autonomous driving: A comprehensive survey. IJCV (2023)

[37] Pang, S., Morris, D., Radha, H.: Clocs: Camera-lidar object candidates fusion for 3d object detection. In: IROS (2020)

[38] Pang, S., Morris, D., Radha, H.: Fast-clocs: Fast camera-lidar object candidates fusion for 3d object detection. In: WACV (2022)

[39] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS (2019)

[40] Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: NeurIPS (2017)

[41] Qin, Z., Wang, J., Lu, Y.: Monogrnet: A geometric reasoning network for monocular 3d object localization. In: AAAI (2019)

[42] 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: CVPR (2020)

[43] Shi, S., Jiang, L., Deng, J., Wang, Z., Guo, C., Shi, J., Wang, X., Li, H.: Pv-rcnn++: Point-voxel feature set abstraction with local vector representation for 3d object detection. IJCV (2022)

[44] Shi, S., Wang, X., Li, H.: Pointrcnn: 3d object proposal generation and detection from point cloud. In: CVPR (2019)

[45] 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 TPAMI (2020)

[46] Shi, W., Rajkumar, R.: Point-gnn: Graph neural network for 3d object detection in a point cloud. In: CVPR (2020)

[47] Shi, X., Ye, Q., Chen, X., Chen, C., Chen, Z., Kim, T.K.: Geometry-based distance decomposition for monocular 3d object detection. In: ICCV (2021)

[48] Sun, J., Chen, L., Xie, Y., Zhang, S., Jiang, Q., Zhou, X., Bao, H.: Disp r-cnn: Stereo 3d object detection via shape prior guided instance disparity estimation. In: CVPR (2020)

[49] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)

[50] Vora, S., Lang, A.H., Helou, B., Beijbom, O.: Pointpainting: Sequential fusion for 3d object detection. In: CVPR (2020)

[51] Wang, C.Y., Liao, H.Y.M., Wu, Y.H., Chen, P.Y., Hsieh, J.W., Yeh, I.H.: Cspnet: A new backbone that can enhance learning capability of cnn. In: CVPR workshops (2020)

[52] Wang, C., Ma, C., Zhu, M., Yang, X.: Pointaugmenting: Cross-modal augmentation for 3d object detection. In: CVPR (2021)

[53] Wang, H., Shi, C., Shi, S., Lei, M., Wang, S., He, D., Schiele, B., Wang, L.: Dsvt: Dynamic sparse voxel transformer with rotated sets. In: CVPR (2023)

[54] Wang, H., Tang, H., Shi, S., Li, A., Li, Z., Schiele, B., Wang, L.: Unitr: A unified and efficient multi-modal transformer for bird's-eye-view representation. In: ICCV (2023)

[55] Wang, T., Zhu, X., Pang, J., Lin, D.: Fcos3d: Fully convolutional one-stage monocular 3d object detection. In: ICCV (2021)

[56] Wang, Y., Guizilini, V.C., Zhang, T., Wang, Y., Zhao, H., Solomon, J.: Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In: Robot Learning (2022)

[57] Wu, H., Wen, C., Shi, S., Li, X., Wang, C.: Virtual sparse convolution for multimodal 3d object detection. In: CVPR (2023)

[58] Xie, Y., Xu, C., Rakotosaona, M.J., Rim, P., Tombari, F., Keutzer, K., Tomizuka, M., Zhan, W.: Sparsefusion: Fusing multi-modal sparse representations for multi-sensor 3d object detection. In: ICCV (2023)

[59] Xu, S., Zhou, D., Fang, J., Yin, J., Bin, Z., Zhang, L.: Fusionpainting: Multimodal fusion with adaptive attention for 3d object detection. In: ITSC (2021)

[60] Yan, J., Liu, Y., Sun, J., Jia, F., Li, S., Wang, T., Zhang, X.: Cross modal transformer via coordinates encoding for 3d object dectection. In: ICCV (2023)

[61] Yan, Y., Mao, Y., Li, B.: Second: Sparsely embedded convolutional detection. Sensors (2018)

[62] Yang, C., Chen, Y., Tian, H., Tao, C., Zhu, X., Zhang, Z., Huang, G., Li, H., Qiao, Y., Lu, L., et al.: Bevformer v2: Adapting modern image backbones to bird's-eye-view recognition via perspective supervision. In: CVPR (2023)

[63] Yang, H., Zhang, S., Huang, D., Wu, X., Zhu, H., He, T., Tang, S., Zhao, H., Qiu, Q., Lin, B., He, X., Ouyang, W.: Unipad: A universal pre-training paradigm for autonomous driving. In: CVPR (2024)

[64] Yang, Z., Sun, Y., Liu, S., Shen, X., Jia, J.: Ipod: Intensive point-based object detector for point cloud. arXiv:1812.05276 (2018)

[65] Yang, Z., Sun, Y., Liu, S., Shen, X., Jia, J.: Std: Sparse-to-dense 3d object detector for point cloud. In: ICCV (2019)

[66] Yang, Z., Chen, J., Miao, Z., Li, W., Zhu, X., Zhang, L.: Deepinteraction: 3d object detection via modality interaction. In: NeurIPS (2022)

[67] Yin, J., Shen, J., Chen, R., Li, W., Yang, R., Frossard, P., Wang, W.: Is-fusion: Instance-scene collaborative fusion for multimodal 3d object detection. In: CVPR (2024)

[68] Yin, T., Zhou, X., Krahenbuhl, P.: Center-based 3d object detection and tracking. In: CVPR (2021)

[69] Yin, T., Zhou, X., Krähenbühl, P.: Multimodal virtual point 3d detection. In: NeurIPS (2021)

[70] 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. arXiv:1906.06310 (2019)

[71] Zhou, Y., Tuzel, O.: Voxelnet: End-to-end learning for point cloud based 3d object detection. In: CVPR (2018)

[72] Zhu, B., Jiang, Z., Zhou, X., Li, Z., Yu, G.: Class-balanced grouping and sampling for point cloud 3d object detection. arXiv preprint arXiv:1908.09492 (2019)

[73] Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159 (2020)

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect the paper's contributions and scope. The claims are clearly stated and are consistent with the theoretical and experimental results presented in the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our method, specifically that using an attention-based approach to interact with the two modalities makes the detection results sensitive to modality loss.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: We provide detailed theoretical statements and formulas along with their descriptions in the paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide a detailed experimental setup in the paper, and the training and testing details are provided in the supplementary material to ensure the reproducibility of our results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
    (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
    (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
    (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
    (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: We release the experimental details in the paper, and the code will be released after the paper is accepted.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide a detailed experimental setup in the paper, and the training and testing details are provided in the supplementary material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide data explanations and statistical methods for obtaining statistical results in the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide hardware computer resources for training and testing.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in our paper complies with NeurIPS ethical standards in all aspects.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss that although our method has good performance, practical applications need to ensure personnel safety.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The model of the paper dos not address the issues mentioned in the guidelines.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We have annotated the cited papers and datasets in our paper.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

A Appendix

A.1 Additional Implementation Details

During training, we adopt a one-stage strategy like DAL [20]. The whole pipeline is trained for a total of 20 epochs with the AdamW optimizer [35] loading from the pre-trained weights from the ImageNet [11] classification task only. Meanwhile, we use CBGS [72] to resample the training data, and the one-cycle learning policy with a maximum learning rate of 2.0×10^{-4}. The batch size is set to 8 on 4 3090 RTX GPUs. We adopt random flipping along both X and Y-axis, the random scaling in [0.95, 1.05], and random rotation in [-π/8, π/8] to augment the LiDAR data, and the random rotation in [-5.4°, 5.4°] and random resizing in [-0.06, 0.44] to augment the images. During evaluation, we test a single model without any data augmentation on a single 3090 RTX GPU.

A.2 Additional Experiments

A.2.1 3D Multi-Object Tracking Experiments

We evaluate our DH-Fusion on the nuScenes tracking benchmark for 3D multi-object tracking (MOT) task. Following ObjectFusion [4], we adopt the same tracking-by-detection algorithm that uses velocity-based closest point distance matching, which is more effective than 3D Kalman filter [9]. For fair comparisons, we report the results of our DH-Fusion-light capable of real-time detection on the nuScenes validation set, as shown in Tab. 6. We find that our DH-Fusion-light outperforms BEVFusion [33] and ObjectFusion [4] by 2.0 pp and 0.6 pp w.r.t. AMOTA. These results demonstrate that our DH-Fusion provides 3D detection boxes of higher quality, benefiting the downstream task of 3D MOT.

Table 6: Comparisons on nuScenes validation set for 3D multi-object tracking.

Methods	AMOTA ↑	AMOTP ↓	IDS ↓
TransFusion [1]	71.8	60.3	694
BEVFusion [33]	72.8	59.4	764
ObjectFusion [4]	74.2	54.3	611
DH-Fusion-light (Ours)	**74.8**	**50.3**	**539**

A.2.2 Evaluation at Different Depths

Since our fusion strategy is depth-aware, it is necessary to validate our method at different depths. Following [4], we categorize annotation and prediction ego distances into three groups: Near (0-20m), Middle (20-30m), and Far (>30m). As shown in Tab. 7, compared to ObjectFusion [4], our DH-Fusion-light consistently improves performance across all depth ranges. Specifically, our method achieves a 47.1 mAP in the long range (>30m), surpassing ObjectFusion by 5.5 pp w.r.t. mAP. These results indicate that our method is more effective across different depths, especially in detecting distant objects.

Table 7: Comparisons on nuScenes validation set at different depths. The numbers are **mAP**.

Methods	Near	Middle	Far
TransFusion-L [1]	77.5	60.9	34.8
BEVFusion [33]	79.4	64.9	40.0
ObjectFusion [4]	79.7	65.4	41.6
DH-Fusion-light (Ours)	**80.3**	**66.5**	**47.1**

A.2.3 Detailed Results on the nuScenes-C

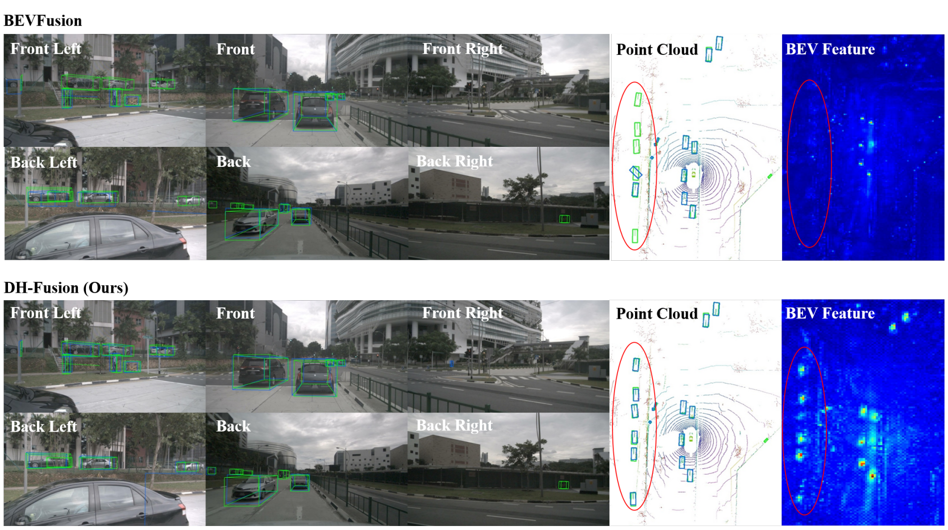
We further provide the detailed results of each fine-grained corruption on nuScenes-C in Tab. 8. The results are highly consistent with the average values of each kind of data corruption.

A.3 More Visualization

As an extension of Fig. 6 in the manuscript, we provide additional examples of 3D object detection results and BEV features from our baseline, BEVFusion [33], and our DH-Fusion. In various samples, our method consistently achieves higher accuracy and recall in 3D detection results, with stronger feature responses for distant objects compared to BEVFusion. These results demonstrate the effectiveness of the proposed method in dynamically adjusting the weights of features based on depth during fusion at both global and local levels.

Table 8: Comparisons for each corruption level on the nuScenes-C. Corruptions exist in both modalities by default. (L) means that only the point cloud modality has corruptions, and (C) means that only the image modality has corruptions. Numbers are **NDS / mAP**.

Corruption		FUTR3D	TransFusion	BEVFusion	**DH-Fusion**
None		68.5 / 64.17	69.82 / 66.38	71.40 / 68.45	**73.30 / 69.75**
Weather	Snow	61.52 / 52.73	68.29 / 63.30	68.33 / 62.84	**71.47 / 65.98**
	Rain	64.47 / 58.40	69.40 / 65.35	70.14 / 66.13	**72.05 / 67.32**
	Fog	61.20 / 53.19	62.62 / 53.67	62.73 / 54.10	**72.13 / 67.24**
	Sunlight	63.61 / 57.70	61.36 / 55.14	68.95 / 64.42	**73.18 / 69.44**
Sensor	Density	67.58 / 63.72	69.42 / 65.77	71.01 / 67.79	**72.94 / 69.15**
	Cutout	66.91 / 62.25	68.30 / 63.66	70.09 / 66.18	**71.99 / 67.45**
	Crosstalk	67.17 / 62.66	68.83 / 64.67	70.72 / 67.32	**73.23 / 69.55**
	FOV Lost	45.66 / 26.32	47.89 / 24.63	**48.65 / 27.17**	43.41 / 20.78
	Gaussian (L)	64.10 / 58.94	62.32 / 55.10	65.99 / 60.64	**69.04 / 63.51**
	Uniform (L)	67.28 / 63.21	68.68 / 64.72	70.18 / 66.81	**72.54 / 68.79**
	Impulse (L)	67.47 / 63.42	69.06 / 65.51	70.63 / 67.54	**72.75 / 68.91**
	Gussian (C)	62.92 / 54.96	68.94 / 64.52	69.35 / 64.44	**71.55 / 66.16**
	Uniform (C)	64.43 / 57.61	69.33 / 65.26	70.06 / 65.81	**72.46 / 67.99**
	Impulse (C)	63.07 / 55.16	68.89 / 64.37	69.25 / 64.30	**71.66 / 66.41**
Motion	Compensation	**39.62 / 31.87**	25.69 / 9.01	36.76 / 27.57	32.51 / 15.99
	Moving Obj.	56.41 / 45.43	60.03 / 51.01	59.42 / 51.63	**68.12 / 60.62**
	Motion Blur	63.44 / 55.99	68.85 / 64.39	69.38 / 64.74	**70.58 / 65.95**
Object	Local Density	67.62 / 63.60	69.34 / 65.65	70.77 / 67.42	**72.48 / 68.87**
	Local Cutout	66.45 / 61.85	67.97 / 63.33	68.11 / 63.41	**69.62 / 64.17**
	Local Gaussian	66.85 / 62.94	67.96 / 63.76	68.32 / 64.34	**71.32 / 67.14**
	Local Uniform	67.92 / 64.09	69.67 / 66.20	70.68 / **67.58**	**71.34** / 66.03
	Local Impulse	67.89 / 64.02	69.64 / 66.29	70.93 / 67.91	**71.83 / 68.15**
	Shear	61.15 / 55.42	66.43 / 62.32	62.95 / 60.72	**68.41 / 65.23**
	Scale	62.00 / 56.79	67.81 / 64.13	66.00 / 64.57	**71.40 / 68.90**
	Rotation	63.67 / 59.64	67.42 / 63.36	66.31 / 65.13	**71.62 / 68.35**
Alignment	Spatial	67.75 / 63.77	69.72 / 66.22	71.35 / 68.39	**71.95 / 69.52**
	Temporal	57.91 / 51.43	54.23 / 43.65	56.62 / 49.02	**62.53 / 55.24**
Average		62.82 / 56.99	64.71 / 58.73	66.06 / 61.03	**68.67 / 63.07**



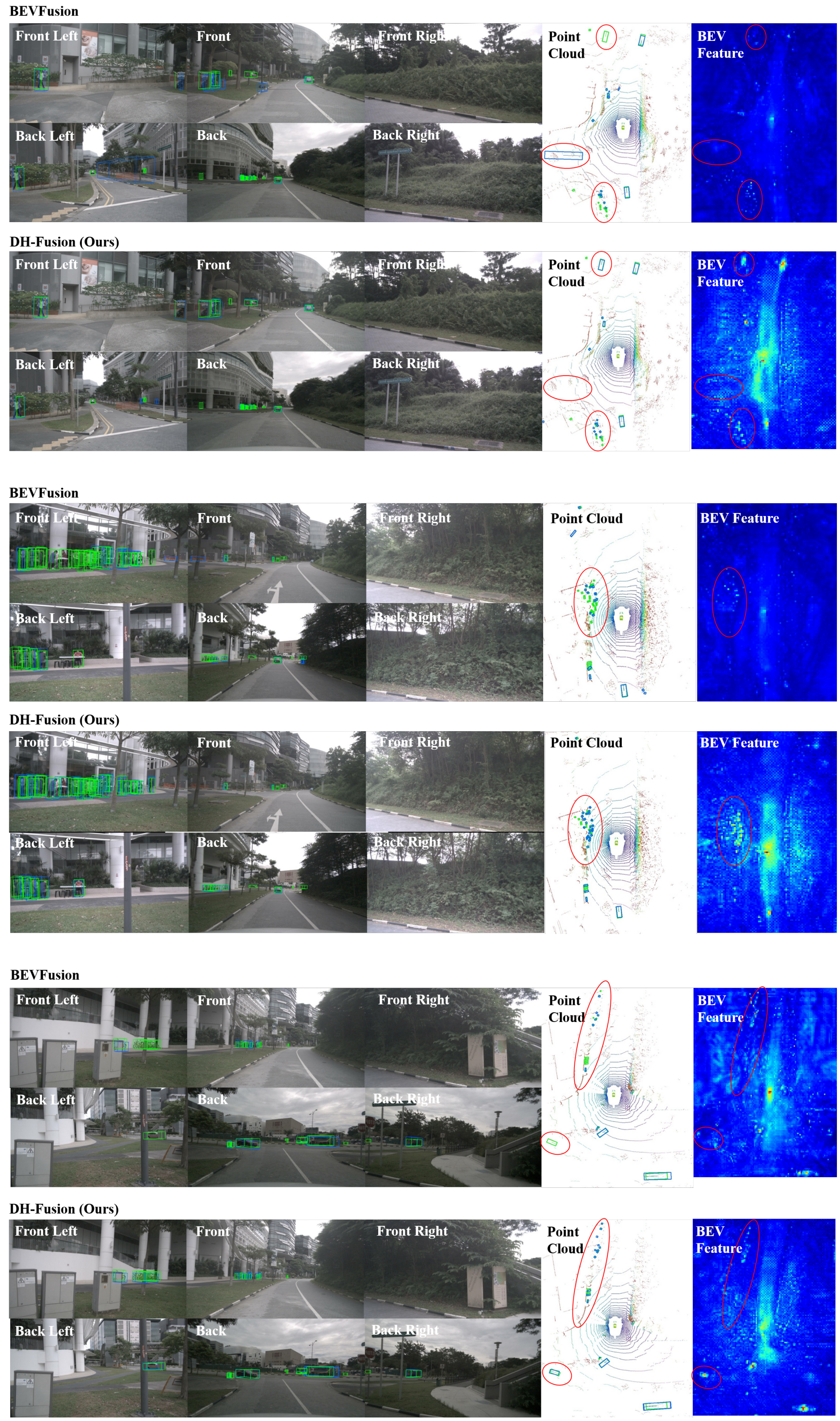


Figure 7: More examples of 3D object detection results and BEV features from BEVFusion and ours. We show the ground truth boxes in green, and the prediction boxes in blue. We use red circles to highlight the comparisons of ours with BEVFusion.