ObjectFusion: Multi-modal 3D Object Detection with Object-Centric Fusion

Qi Cai, Yingwei Pan, Ting Yao, Chong-Wah Ngo and Tao Mei
University of Science and Technology of China
HiDream.ai Inc
Singapore Management University
{cqcaiqi, panyw.ustc, tingyao.ustc}@gmail.com, cwngo@smu.edu.sg, tmei@hidream.ai

Abstract

*Recent progress on multi-modal 3D object detection has featured BEV (Bird-Eye-View) based fusion, which effectively unifies both LiDAR point clouds and camera images in a shared BEV space. Nevertheless, it is not trivial to perform camera-to-BEV transformation due to the inherently ambiguous depth estimation of each pixel, resulting in spatial misalignment between these two multi-modal features. Moreover, such transformation also inevitably leads to projection distortion of camera image features in BEV space. In this paper, we propose a novel Object-centric Fusion (**ObjectFusion**) paradigm, which completely gets rid of camera-to-BEV transformation during fusion to align object-centric features across different modalities for 3D object detection. ObjectFusion first learns three kinds of modality-specific feature maps (i.e., voxel, BEV, and image features) from LiDAR point clouds and its BEV projections, camera images. Then a set of 3D object proposals are produced from the BEV features via a heatmap-based proposal generator. Next, the 3D object proposals are reprojected back to voxel, BEV, and image spaces. We leverage voxel and RoI pooling to generate spatially aligned object-centric features for each modality. All the object-centric features of three modalities are further fused at object level, which is finally fed into the detection heads. Extensive experiments on nuScenes dataset demonstrate the superiority of our ObjectFusion, by achieving 69.8% mAP on nuScenes validation set and improving BEVFusion by 1.3%.*

1. Introduction

3D object detection is one of the fundamental tasks in 3D vision, which aims to localize the objects of interest in the 3D scene. This task plays a critical role in perceiving the surrounding environment of autonomous driving. For robust and high-quality detection, the current practice mostly follows multi-sensor fusion paradigm, which integrates the data derived from different sensors (e.g., cameras and LiDAR). On one hand, the RGB images convey rich texture and semantics of objects captured from different angles of cameras. LiDAR, on the other hand, observes the environment by emitting pulses of light, yielding point cloud data that preserves accurate geometry information regardless of lighting conditions. As RGB images are vulnerable to lighting conditions, LiDAR point clouds naturally complement camera images and lead to the idea of blending these multi-sensor data for robust and accurate perception.

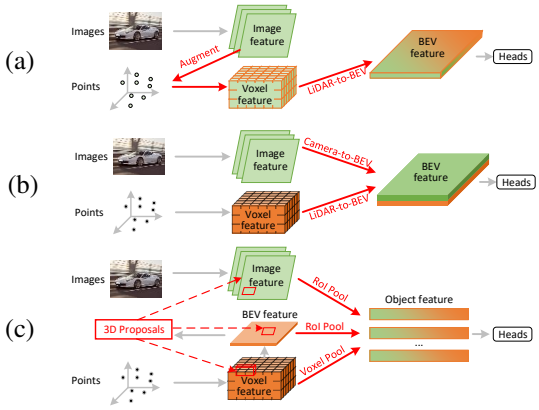

Figure 1: Illustration of (a) point-based fusion [48], (b) BEV-based fusion [34], and (c) our object-centric fusion.

Considering that camera and LiDAR offer different perspectives (i.e., images versus point clouds) of 3D scenes, the mainstream approaches unify them into a shared representation space. Cross-modal projection, such as by point-based [48, 49] or BEV-based [30, 34] fusing has been proposed. The point-based fusion strategy [48, 49] first builds the correspondence between 3D points and image pixels via calibration matrices. As shown in Figure 1(a), the images are projected into the raw point space, augmenting points with the corresponding image features or semantic scores. The augmented points are further transformed into BEV features for 3D detection. However, this point-based fusion only associates points with a small portion of images, leaving the rich semantic information of images under-exploited. Instead, BEV-based fusion [30, 34] projects both the im-

ages and point clouds into a shared BEV space through camera-to-BEV and LiDAR-to-BEV transformations, leading to augmented BEV features for object detection (Figure 1(b)). Despite showing encouraging performances, BEV-based fusion heavily relies on the off-the-shelf depth estimator (e.g., LSS [39]) to estimate the depth of each image pixel for camera-to-BEV transformation. As pointed out in BEVDepth [24], the estimation is error-prone and not trivial. Any inaccurate depth estimation will result in spatial misalignment between image pixels and points within the shared BEV space, which subsequently affects object detection. Moreover, recall that image and BEV features reflect two different data peculiarities: images are captured from different perspective views, while BEV features are formulated as top-down aggregation along height dimension. Hence, directly projecting the image features into BEV space will inevitably lead to projection distortion and destroy the original semantic structures within images.

In view of the limitations of point-based and BEV-based fusion strategies, *is it possible to perform multi-modal fusion without requiring the non-trivial inter-modality transformation (e.g., camera-to-BEV projection)?* We address the challenge by presenting a unique fusion paradigm, named Object-centric Fusion (ObjectFusion), as conceptually depicted in Figure 1(c). Our launching point is to introduce the object-centric representation in each modality, and spatially align the representations according to the 2D/3D bounding box of an object. ObjectFusion is henceforth able to safely unify the object-centric representations of different modalities by eliminating inter-modality transformation during fusion. That is, our ObjectFusion nicely preserves the primary feature of each modality, and enables multi-modal fusion at object level with better spatial alignment.

ObjectFusion first generates two modality-specific feature maps (voxel and image features) from point clouds and images via regular 3D/2D networks. The sparse voxel feature maps are flattened along the height dimension, leading to denser BEV features. ObjectFusion leverages a heatmap-based proposal generator to estimate the objectness score in each position of the BEV features and select the top-ranked positions as initial object queries, which triggers the generation of a set of 3D object proposals. Such 3D proposals are projected into voxel, image, and BEV spaces to align object-centric features in different spaces. Specifically, the object features corresponding to a proposal are generated by voxel pooling [12] or RoI Align [17] in their respective spaces. With the proposals, the features from the three modalities can be effortlessly aligned, without the non-trivial inter-modality transformations as adopted by [30, 34]. ObjectFusion contextualizes the object-centric features with a modality-specific contextual encoder. These features are further concatenated and fed into the detection heads for proposal classification and regression.

2. Related Work

LiDAR-based Approaches. Modern autonomous vehicles are usually equipped with LiDARs and researchers have developed various 3D detection frameworks solely upon point cloud data from LiDARs. Due to the irregular structure of point clouds, a natural solution is to convert the point cloud into grid representations such as range images, pillars, or voxels. For example, Lasernet [35], the pioneering work on range images, exploits a fully convolutional network to predict distributions over 3D boxes for each point and then fuses these distributions for 3D predictions. Later on, [2, 8, 14] proceed in this direction and design customized convolutional kernels for range image processing. PointPillar [22] organizes the points in each vertical column as a pillar and uses a PointNet [41] to learn each pillar's feature. VoxelNet [63] introduces an end-to-end framework where the space is divided into grid voxels, which are further encoded via 3D convolutional network. Later works [55, 23] improve the performance of VoxelNet by designing more effective voxel encoding strategies. Some other approaches [28, 43, 44, 56, 57] directly operate on the points to avoid quantization errors. Recent research starts to focus on designing novel detection heads, e.g., anchor-free heads [9, 15, 60] and Transformer decoder heads [7, 13, 36].

Camera-based Approaches. Similar to the approach taken in 2D detection [5, 6, 16, 42] from images, the task of 3D detection can also be executed within the domain of image space. For this direction, earlier works are commonly established based on 2D counterparts by predicting 3D boxes from 2D proposals [3, 37, 45, 54]. Similar to FCOS [46] for the image domain, FCOS3D [51] directly regresses 3D bounding boxes and class scores from object features. Later on, PGD [50] introduces a geometric relation graph across objects to facilitate depth estimation, which further improves the quality of 3D object detection. Another line of research utilizes Transformer [47] decoder to interact learnable 3D object queries with 2D image features. In particular, DETR3D [52] and PETR [32] extract 2D features from camera images and then use 3D object queries to index these 2D features. Recently, the mainstream approaches start to convert image features into BEV space and then perform detection with LiDAR-based approaches. BEVDet [18] adopts a view Transformer [39] to render virtual point clouds with image features, which are further pooled to BEV features. BEVFormer [29] designs a spatiotemporal Transformer to generate BEV features by exploiting both spatial and temporal clues.

Fusion-based Approaches. This direction explores the complementary among multi-sensor data to boost 3D object detection with multi-modal fusion at different levels. Early works [10, 21, 40] fuse the LiDAR and camera features at the proposal level. MV3D [10] and AVOD [21]

learn 3D object proposals from the point clouds and then combine region-wise features from multiple views to predict 3D boxes. However, due to the limited capacity in proposal generation and 3D feature learning, the performances of proposal-level fusion greatly fall behind recent point-based [19, 48, 49, 61, 53] and BEV-based fusion methods [30, 34]. PointPainting [48] and PointAugmenting [49] decorate the source points with semantic scores or features extracted from 2D images, which can be used in any LiDAR-based method for 3D detection. With recent developments in BEV perception, transforming images and point clouds into the shared BEV space emerges as a popular fusion strategy. BEVFusion [30, 34] explicitly predicts depth distribution of each pixel and scatters image feature to BEV space with BEV pooling. There are also some works [26, 62] that focus on voxel-based fusion and training strategies etc.

Our work also falls into the category of fusion-based approaches. Unlike recent point-level or feature-level fusion methodologies that hinge on non-trivial inter-modality projections, our ObjectFusion uniquely integrates high-quality object-centric representations across three modalities without necessitating inter-modality projection. Such design enables a feasible multi-modal fusion at object level with precise spatial alignment across different modalities.

3. Approach

This section presents the architecture of our proposed Object-centric Fusion (ObjectFusion) paradigm, along with the design of each key component. As shown in Figure 2, ObjectFusion is composed of three main components: 1) modality-specific encoders that learn primary voxel/BEV/image representations for each modality, 2) object-centric fusion module that unifies three object-centric representations in different modalities, and 3) detection heads to predict 3D boxes and classes. Specifically, given LiDAR point cloud and the corresponding multi-view camera images, the points encoder and image encoder first extract voxel and image feature maps, respectively. The voxel feature maps are then flattened along Z-axis to produce BEV feature maps. In the object-centric fusion module, a set of 3D proposals are generated based on the BEV feature map. For each proposal, we project the 3D boxes into voxel, BEV, and image spaces, and extract object-centric features from corresponding modality-specific feature maps. The object-centric features in each modality are further contextually encoded via modality-specific context encoders to fully exploit the inter-object relations. Subsequently, the object-centric features from different modalities are concatenated and fed into the detection heads.

3.1. Modality-Specific Encoders

Given the multi-modal inputs of point cloud and image data, three modality-specific encoders are devised to extract the primary modality-specific features in voxel, BEV, and image spaces. Formally, the input point cloud consists of a set of N_p points: $P = \{p_i | i \in [1, N_p]\}$. Each point is represented as a 4-dimensional vector $p = (x, y, z, r)$, where x, y, z is the coordinates along X-axis, Y-axis, and Z-axis, and r is the reflection intensity. The multi-view images contain N_c images $I = \{I_n | I_n \in R^{3\times H\times W}; n \in [1, N_c]\}$, and I_n denotes the image captured by the n-th camera.

Voxel Encoder. For point cloud P, we leverage the popular feature extraction module VoxelNet [63] to learn regular voxel representation. In particular, the points are divided into equal-spaced voxels and the point coordinates in the same voxel are aggregated as a voxel feature. In this way, the irregular points are converted into grid voxels and each voxel is accompanied by a feature vector. Then a 3D backbone stacks multiple sparse convolutional layers to extract voxel features $F_{\mathcal{V}} \in R^{N_{\mathcal{V}}\times C_{\mathcal{V}}}$ and voxel centers $V_{\mathcal{C}} \in R^{N_{\mathcal{V}}\times 3}$ of the point cloud. Here $N_{\mathcal{V}}$ is the number of voxels and $C_{\mathcal{V}}$ is the voxel feature channel.

BEV Encoder. On the basis of the voxel features $F_{\mathcal{V}}$ and voxel centers $V_{\mathcal{C}}$, we stack the features along Z-axis to compress voxel features at the same X-Y coordinates into a single feature vector. In this way, the voxel feature is converted to a 2D feature map and we further utilize a 2D convolutional network to extract BEV features $F_{\mathcal{B}} \in R^{C_{\mathcal{B}}\times H_{\mathcal{B}}\times W_{\mathcal{B}}}$, where $C_{\mathcal{B}}$ is the channel number of the BEV feature.

Image Encoder. For each image I_n, we use the Swin Transformer [33] as 2D backbone to extract multi-scale image feature maps, and additionally employ FPN [31] to fuse multi-scale feature maps into a single-scale feature map $F_{\mathcal{I}_n} \in R^{C_{\mathcal{I}}\times H_{\mathcal{I}}\times W_{\mathcal{I}}}$. The feature map is downsampled at 1/8 of original image resolution, i.e., $H_{\mathcal{I}} = H/8$ and $W_{\mathcal{I}} = W/8$. We stack the feature maps from N_c images to form the whole image feature map $F_{\mathcal{I}} \in R^{N_c\times C_{\mathcal{I}}\times H_{\mathcal{I}}\times W_{\mathcal{I}}}$.

3.2. Object-Centric Fusion

In an effort to unify multi-modal feature maps, one dominant force in recent advances is to use BEV-based fusion to project point and image features into a shared BEV space through LiDAR-to-BEV/camera-to-BEV transformation. However, the camera-to-BEV transformation hinges on pre-learned depth estimator to obtain the inherently ambiguous depth estimation of each pixel. Any inaccurate depth estimation can potentially result in spatial misalignment between the image and point feature maps. In addition, considering the fundamentally different perspective views of images and BEV features, directly projecting image features into BEV space will lead to projection distortion. To alleviate these issues, we design a novel object-centric fusion module by unifying the object-centric representation in each modality at the object level, without the requirement of inter-modality projection. First, we generate a set of 3D object proposals from BEV features by using a

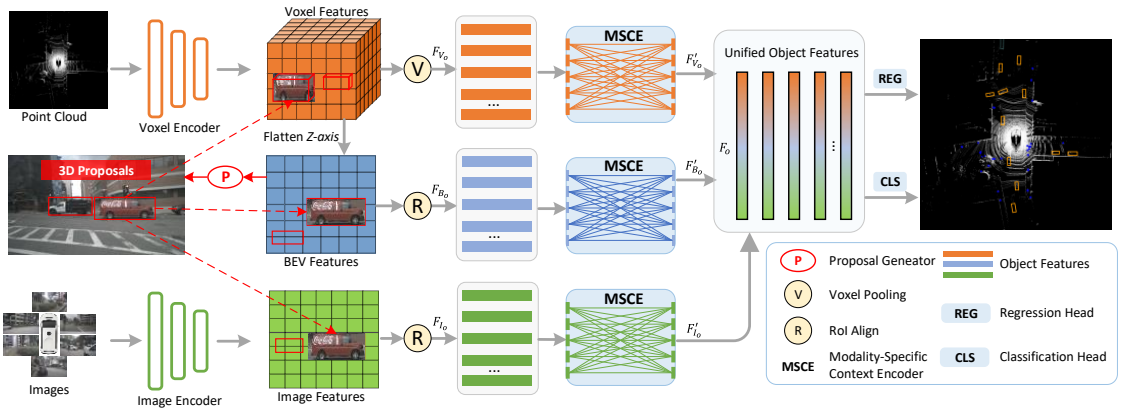

Figure 2: An overview of our ObjectFusion framework. The input point cloud and images are separately processed by two modality-specific encoders (voxel encoder and image encoder) to generate voxel and image features. In addition, the voxel features are further flattened along Z-axis to produce BEV features. After that, a set of 3D object proposals are generated based on image-augmented BEV features through a heatmap-based proposal generator. These 3D proposals are projected back to voxel, BEV, and image space, yielding object-centric features in each space via voxel pooling or RoI Align. Next, the object-centric features are contextually encoded with a modality-specific context encoder, which are finally concatenated at object level and fed into regression/classification heads for detection.

heatmap-based proposal generator. Then we project the 3D object proposals into voxel, BEV and image spaces, and extract object-centric features in each space. With a two-stage scheme, the object-centric features are first contextualized with their respective encoders and then concatenated into fused object features for each proposal.

Heatmap-based Proposal Generator. The proposal generator takes image-augmented BEV features as input to generate preliminary 3D proposals, which facilitates further object-centric feature extraction and fusion. Note that we employ BEVFusion [34] to augment BEV features with additional image information, which guarantees that most true positives are in the candidate pool for the latter stage of 3D detection head. Conditioned on the augmented BEV features, we first leverage a heatmap head [1] to predict a class-specific objectness map $S \in R^{K \times H_{\mathcal{B}} \times W_{\mathcal{B}}}$, where K is the number of interested object classes. Then we select top-O positions in S with the highest objectness scores, which can be regarded as the positions of initial 3D object queries $\{q_o^p | q_o^p \in R^2; o \in [1, O]\}$. To eliminate redundant proposals clustered at the same object, we use the peak finding algorithm to find the local maxima at each objectness map position when selecting the top-O positions. Then the query features $\{q_o^f | q_o^f \in R^{C_{\mathcal{B}}}; o \in [1, O]\}$ are initialized from the BEV features $F_{\mathcal{B}}$ at the corresponding positions. Subsequently, we use a Transformer decoder layer to aggregate relevant BEV features $F_{\mathcal{B}}$ into object query features q_o^f. Furthermore, the object queries are decoded into 3D proposals $B = \{b_o | b_o = (x, y, z, w, l, h, \theta); o \in [1, O]\}$ independently through a feed-forward network, where (x, y, z) is the center and (w, l, h, θ) are the width, length, height, and yaw angle respectively.

Object-Centric Voxel Feature. Voxel feature is a natural regular representation of point cloud data, where each voxel represents a grid-size 3D space. The benefits of voxel features lie in the encoding of precise localization and geometric information. Accordingly, given voxel features $F_{\mathcal{V}}$, voxel centers $V_{\mathcal{C}}$, and the 3D bounding box b_o of each proposal, we capitalize on voxel pooling [12] to extract the object-centric voxel feature. Concretely, the voxel pooling starts by dividing b_o into $G \times G \times G$ equal-spaced sub-voxels and the center point of each sub-voxel is regarded as the grid point. Next, for each grid point, we seek the nearby voxels in $C_{\mathcal{V}}$ within a pre-defined radius and integrate corresponding voxel features from $F_{\mathcal{V}}$ into the grid points. Finally, the features of all grid points are concatenated together to form the primary object-centric voxel feature $\hat{F}_{V_o} \in R^{C_{\mathcal{V}} \times G \times G \times G}$ for proposal b_o. Such process can be formally denoted as:

$$\hat{F}_{V_o} = \text{VoxelPooling}(F_{\mathcal{V}}, V_{\mathcal{C}}, b_o). \quad (1)$$

Based on the primary feature $\hat{F}_{V_o}$, we further apply a 3D convolutional layer with global average pooling to output object-centric voxel feature $F_{V_o} = \text{3DConv}(\hat{F}_{V_o})$.

Object-Centric BEV Feature. Compared to the sparse voxel features $F_{\mathcal{V}}$, BEV features $F_{\mathcal{B}}$ belong to another type of point cloud representation which is much denser with richer context information via aggregation. Since the BEV features are represented as a 2D feature map in the shape of $R^{C_{\mathcal{B}} \times H_{\mathcal{B}} \times W_{\mathcal{B}}}$, we take the inspiration from 2D RoI pooling and adopt RoIAlign [17] to extract object-centric features from BEV feature map. Technically, the eight corners of the 3D bounding box b_o are first projected to BEV space by ignoring the height dimension. Then a minimum axis-aligned bounding box $b_o^{\mathcal{B}} \in R^4$ which can cover all eight concerns in BEV space is taken as the projection of b_o. Next, the RoIAlign divides $b_o^{\mathcal{B}}$ into $r \times r$ equal-spaced sub-regions and utilizes bilinear interpolation to aggregate relevant features from $F_{\mathcal{B}}$ into each sub-region. Finally, the features

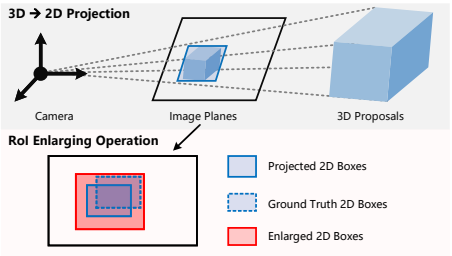

Figure 3: An illustration of RoI enlarging operation during the projection from 3D proposals to 2D boxes.

from all sub-regions are concatenated to construct the primary object-centric BEV feature $\hat{F}_{B_o} \in R^{C_\mathcal{B} \times r \times r}$:

$$\hat{F}_{B_o} = \text{RoIAlign}(F_\mathcal{B}, b_o). \tag{2}$$

Similar to the object-centric voxel features, we adopt a convolution layer with global pooling to transform $\hat{F}_{B_o}$ into the output object-centric BEV feature $F_{B_o} = \text{2DConv}(\hat{F}_{B_o})$.

Object-Centric Image Feature. Both voxel and BEV features excel at capturing the geometric information of objects, while lacking the ability to model object texture and appearance. Instead, the image feature contains rich texture and appearance information of objects. Targeting for learning object-centric features from images, we project the 3D proposal b_o into the camera image plane, and obtain the 2D bounding box $b_o^\mathcal{I} \in R^4$ by using calibration matrices. In particular, we compute the eight corners of b_o and their projected X-Y coordinates on the images. Considering that multiple images are provided in different camera views, we need to decide which camera to use for object-centric feature extraction. Depending on the positions and sizes of objects, the projection of a 3D box could fall into multiple camera Field of Views (FoVs) or outside of all camera FoVs. If the projected corners are outside of all camera FoVs, we discard the image feature for b_o. Otherwise, we select the image which covers the most projected corners to extract object-centric features. Here we compute the minimum axis-aligned bounding box (AABB) on the selected image plane. The intersection between the minimum AABB and image boundary is taken as the 2D bounding box $b_o^\mathcal{I}$ to learn object-centric features $\hat{F}_{I_o}$. Considering that the projections between 3D and 2D spaces are not perfect due to sensor misalignment, we adopt a simple yet effective RoI enlarging operation to alleviate such calibration error. As shown in Figure 3, our RoI enlarging operation strategy doubles the RoI sizes of projected 2D boxes on image planes. This way ensures that the object features extracted from images can still encompass interested objects even when 3D and 2D spaces are not perfectly aligned. Finally, we adopt RoIAlign followed by convolutional layers plus global pooling to extract the object-centric image feature F_{I_o} in image space.

Two-stage Fusion Scheme. Next, given the object-centric features from voxel, BEV, and image spaces, we design a two-stage fusion scheme for cross-object and cross-modal representation learning. In the first stage, an object feature is contextualized with the features of remaining objects in the same modality via a Modality-Specific Context Encoder (MSCE). Transformers [25, 27, 47, 58, 59] have been widely used in the context modeling of different modality features. Therefore, the MSCE is implemented as a single-layer Transformer encoder. In this way, we can obtain the enhanced object-centric features in each modality:

$$\{F'_{V_o} | o \in [1, O]\} = \text{MSCE}_V(\{F_{V_o} | o \in [1, O]\}), \tag{3}$$

$$\{F'_{B_o} | o \in [1, O]\} = \text{MSCE}_B(\{F_{B_o} | o \in [1, O]\}), \tag{4}$$

$$\{F'_{I_o} | o \in [1, O]\} = \text{MSCE}_I(\{F_{I_o} | o \in [1, O]\}). \tag{5}$$

In the second stage, for each object proposal b_o, we concatenate the corresponding enhanced object-centric features in three modalities, and embed them with a feed-forward network (FFN) to achieve the unified object-centric feature F_o:

$$F_o = \text{FFN}(\text{Concat}(F'_{V_o}, F'_{B_o}, F'_{I_o})). \tag{6}$$

Such unified object-centric feature F_o is further integrated with the query feature q_o^f, which will be fed into the following detection heads. Accordingly, the overall two-stage fusion scheme operates as follows:

$$\hat{q}_o^f = \text{FFN}(\text{Concat}(F_o, q_o^f)) + q_o^f. \tag{7}$$

3.3. Detection Heads

Based on the upgraded query features $\{\hat{q}_o^f | o \in [1, O]\}$ that contain rich multi-modal contextual information of objects, we leverage a decoder-based detection module to predict the object class and 3D bounding box. Specifically, a Transformer-based decoder first embeds each query feature $\hat{q}_o^f$ by cross-attending to each other. Note that the design of detection module is similar to the one in BEVFusion [34]. However, instead of attending to both query and BEV features in BEVFusion, we only attend to query features since the query features are already strengthened with additional multi-modal information. Next, a regression head and classification head are applied to decode the query features into 3D boxes and object classes. Both heads are implemented as two-layer feed-forward networks.

4. Experiments

4.1. Experimental Setup

Dataset and Metric. We evaluate ObjectFusion on the challenging large-scale nuScenes dataset [4], which is collected with a 32-beam LiDAR and six cameras. The six images cover 360-degree surroundings and the dataset provides calibration matrices that enable precise projection from 3D points to 2D pixels. We use the mAP and NDS across all categories as the primary metrics for evaluation following [1, 34]. Note that NDS metric is a weighted average of mAP and other breakdown metrics (e.g.,translation, scale, orientation, velocity, and attributes errors).

Table 1: Comparisons with state-of-the-art methods on nuScenes validation and test set for 3D object detection. The Modality column: "L" = only LiDAR data; "LC" = the use of both LiDAR and camera data. The Fusion column: "P" = Point-based fusion; "B" = BEV-based fusion; "O" = Object-centric fusion; "✗" = Not applicable. "†": performances from BEVFusion [34]. "‡": we use the released model to calculate per-category AP. The best performances are marked with **bold** font.

Method	Modality	Fusion	mAP(%)	NDS(%)	Car	Truck	C.V.	Bus	Trailer	Barrier	Motor.	Bike	Ped.	T.C.
*Performances on **validation set**:*														
TransFusion-L [1]	L	✗	65.1	70.1	86.5	59.6	25.4	74.4	42.2	74.1	72.1	56.0	86.6	74.1
PointPainting [48]†	LC	P	65.8	69.6	-	-	-	-	-	-	-	-	-	-
TransFusion [1]	LC	B	67.3	71.2	87.6	62.0	27.4	75.7	42.8	73.9	75.4	63.1	87.8	77.0
BEVFusion [30]	LC	B	67.9	71.0	88.6	65.0	28.1	75.4	41.4	72.2	76.7	**65.8**	88.7	76.9
BEVFusion [34]‡	LC	B	68.5	71.4	89.2	64.6	30.4	75.4	42.5	72.0	78.5	65.3	88.2	79.5
ObjectFusion	LC	O	**69.8**	**72.3**	**89.7**	**65.6**	**32.0**	**77.7**	**42.8**	**75.2**	**79.4**	65.0	**89.3**	**81.1**
*Performances on **test set**:*														
PointPillar [22]	L	✗	40.1	55.0	76.0	31.0	11.3	32.1	36.6	56.4	34.2	14.0	64.0	45.6
CenterPoint [60]	L	✗	60.3	67.3	85.2	53.5	20.0	63.6	56.0	71.1	59.5	30.7	84.6	78.4
TransFusion-L [1]	L	✗	65.5	70.2	86.2	56.7	28.2	66.3	58.8	78.2	68.3	44.2	86.1	82.0
PointPainting [48]	LC	P	46.4	58.1	77.9	35.8	15.8	36.2	37.3	60.2	41.5	24.1	73.3	62.4
PointAugmenting [49]	LC	P	66.8	71.0	87.5	57.3	28.0	65.2	60.7	72.6	74.3	50.9	87.9	83.6
MVP [61]	LC	P	66.4	70.5	86.8	58.5	26.1	67.4	57.3	74.8	70.0	49.3	89.1	85.0
TransFusion [1]	LC	B	68.9	71.7	87.1	60.0	33.1	68.3	60.8	78.1	73.6	52.9	88.4	86.7
BEVFusion [30]	LC	B	69.2	71.8	88.1	**60.9**	34.4	69.3	62.1	78.2	72.2	52.2	89.2	85.2
BEVFusion [34]	LC	B	70.2	72.9	88.6	60.1	39.3	69.8	**63.8**	**80.0**	74.1	51.0	89.2	86.5
ObjectFusion	LC	O	**71.0**	**73.3**	**89.4**	59.0	**40.5**	**71.8**	63.1	76.6	**78.1**	**53.2**	**90.7**	**87.7**

Implementations. We implement the proposed ObjectFusion in the PyTorch [38] framework, based on the open-source MMdetection3D [11] and BEVFusion [34] codebases. For the voxel encoder, we use VoxelNet [63] as the backbone and a SECOND [55] 2D network is adopted to obtain BEV features. The voxel size is set as [0.075m, 0.075m, 0.1m], and the point cloud range is [-54m, -54m, -3m, 54m, 54m, 5m] in X, Y, and Z-axis, respectively. For the image encoder, we use the Swin-T [33] network as the backbone and FPN [17] to fuse multi-scale feature maps. The resolution of input images is resized and cropped to 256×704 as in BEVFusion. The image backbone is pre-trained on the nuImage [4] dataset for 2D detection and the neck is randomly initialized. The MSCE is implemented as a single layer Transformer encoder with 8 heads and the FFN in object-centric fusion contains two layers MLP with the hidden dimension of 128. During training, we adopt a two-stage strategy. In the first stage, we train the LiDAR branch without using images for 20 epochs. Then we initialize the overall multi-modal fusion model with the pre-trained LiDAR branch weights and continue training for another 6 epochs. For both stages, we utilize random flip, random rotation in $[-\pi/4, \pi/4]$, random translation with std=0.5, and random scaling in $[0.9, 1.1]$ to augment the LiDAR data. We use CBGS [64] to resample the training data. For the first 15 epochs, we add copy-paste data augmentation [55] to reduce overfitting. For the second stage, we additionally use random rotation in $[-5.4°, 5.4°]$ and random resizing in $[0.38, 0.55]$ to augment the images. Following common practice [1, 34], we align the previous nine LiDAR sweeps into the current frame for a denser point cloud. During training, we use Adam optimizer [20] with one-cycle learning rate policy, where the maximum learning rate is 0.001 and the weight decay is 0.01. The batch size is set as 16/8 for the first/second stage. For all runs, we use four NVIDIA V100 16G GPUs for training. At inference, we remove the data augmentation and set batch size to 1.

4.2. Comparisons with State-of-the-Art Methods

3D Object Detection. We first compare the performances of our ObjectFusion and other state-of-the-art approaches on the nuScenes validation set for 3D object detection task. As shown in Table 1, ObjectFusion establishes to-date the best performances on validation set (69.8% in mAP and 72.3% in NDS), which consistently outperform all single-modality and multi-modal fusion approaches. In general, by exploiting the complementary information among different modalities, the multi-modal fusion series ("LC" in Modality column) exhibit better performances than the single-modality series ("L" in Modality column). Specifically, the point-based fusion (PointPainting [48]) directly augments points with a small portion of projected image semantic scores. ObjectFusion preserves the primary features in each modality and exploits rich object-level image features for fusion, without losing substantial image semantics, leading to a large absolute performance gain of 4.0% mAP. Furthermore, when compared to the BEV-based fusion (i.e., TransFusion [1] and BEVFusion [34]), ObjectFusion achieves the absolute performance improvement of 2.5% and 1.3% in mAP, respectively. In particular, TransFusion performs BEV-based fusion by measuring cross-attention between BEV features (query) and the whole image features (keys/values). Such cross-attention progress directly refines BEV features by aggregating all image features without the consideration of the spatial alignment. As an alternative, our ObjectFusion elegantly fuses the spa-

Table 2: Comparisons with state-of-the-art approaches on nuScenes validation set for 3D multi-object tracking.

Methods	Modality	AMOTA (%) ↑	AMOTP (%)↓	IDS↓
CenterPoint [60]	L	63.7	60.6	640
TransFusion-L [1]	L	69.9	59.9	821
TransFusion [1]	LC	71.8	60.3	794
BEVFusion [34]	LC	72.8	59.4	764
ObjectFusion	LC	**74.2**	**54.3**	**611**

tially aligned multi-modal features at object level, leading to the performance boosts. Moreover, BEVFusion transforms image features into a shared BEV space via camera-to-BEV transformation, which might result in spatial misalignment and projection distortion. In contrast, ObjectFusion manages to eliminate above issues through object-centric fusion without camera-to-BEV transformation in 3D detection head, thereby achieving the best performances.

We also submitted the detection results on nuScenes test set to the official evaluation server, and Table 1 summarizes the performance comparisons. Similar to the observations on the validation set, our ObjectFusion again surpasses all the published multi-modal fusion techniques.

3D Multi-Object Tracking. Next, we further evaluate our ObjectFusion on the nuScenes tracking benchmark for 3D multi-object tracking (MOT) task. Following TransFusion [1], we adopt the same tracking-by-detection algorithm, which directly links objects between consecutive frames greedily. For fair comparisons, we report single model performances without test-time augmentation and model ensembling on nuScenes validation set. As shown in Table 2, ObjectFusion outperforms TransFusion [1] and BEVFusion [34] by 2.4% and 1.4% performance gains in AMOTA metric, which basically validates the generalizability of our ObjectFusion on 3D MOT task.

4.3. Detection Robustness Analysis

In this section, we present the robustness analysis of ObjectFusion which is crucial for practical applications. Robustness is measured by assessing the performance under different lighting and weather conditions, different ego distances and object sizes, and calibration errors. All experiments are conducted on the nuScenes validation set.

Robustness to Lighting and Weather Conditions. Different lighting and weather conditions make the 3D object detection task challenging in practice. For example, the night and fog will make objects more difficult to be captured with cameras due to the poor lighting conditions. Here we follow BEVFusion [34] to evaluate ObjectFusion under different lighting and weather conditions. We split the scenes in validation set into Sunny/Rainy/Day/Night by searching "rain" and "night" keywords in the description of each scene. As shown in Table 3, CenterPoint [60] which only uses LiDAR point cloud is sensitive to rainy weather with

Table 3: Performance comparisons on nuScenes validation set under different lighting and weather conditions.

mAP(%)	Modality	Sunny	Rainy	Day	Night
CenterPoint [60]	L	62.9	59.2	62.8	35.4
BEVDet [18]	C	32.9	33.7	33.7	13.5
BEVFusion [34]	LC	68.2	69.9	68.5	42.8
ObjectFusion	LC	**69.8**	**70.1**	**69.8**	**46.0**

Table 4: Performance comparisons on nuScenes validation set with different ego distances and object sizes.

mAP(%) with different ego distances:				
Distances	Modality	*Near*	*Middle*	*Far*
TransFusion-L [1]	L	77.5	60.9	34.8
BEVFusion [34]	LC	79.4	64.9	40.0
ObjectFusion	LC	**79.7**	**65.4**	**41.6**
mAP(%) with different object sizes:				
Sizes	Modality	*Small*	*Moderate*	*Large*
TransFusion-L [1]	L	44.7	54.5	60.4
BEVFusion [34]	LC	50.3	58.7	64.0
ObjectFusion	LC	**53.0**	**60.7**	**65.0**

3.7% mAP drop compared to sunny weather. And BEVDet [18] which only relies on camera image is severely vulnerable to poor lighting conditions at night with only 13.5% mAP. The results basically demonstrate that neither LiDAR point clouds nor camera images are sufficient for robust 3D object detection. Instead, by integrating both LiDAR point clouds and camera images via BEV-based fusion, BEVFusion [34] significantly boosts up the performances under all conditions. By getting rid of camera-to-BEV transformation and enabling spatially aligned object-centric fusion of multi-modal data, ObjectFusion attains the best performances under each challenging lighting and weather condition. Especially for night scenarios, ObjectFusion outperforms BEVFusion by 3.2% where depth estimation is more challenging for BEVFusion under poor lighting conditions.

Robustness to Ego Distances and Object Sizes. The performances of 3D object detection are commonly sensitive to ego distances (the distances to ego vehicle) and object sizes. Generally, it is difficult to observe distant and small objects from LiDAR sensor. We categorize annotation and prediction ego distances simultaneously into three groups: *Near* (0-20m), *Middle* (20-30m) and *Far* (>30m). We also summarize the object size distributions for each category and define three size levels with equal proportions: *Small*, *Moderate* and *Large*. As shown in Table 4, the LiDAR-only TransFusion-L [1] is sensitive to the change of ego distances and object sizes: 77.5% mAP v.s. 34.8% mAP for *Near* v.s. *Far* objects; 60.4% v.s. 44.7% for *Large* v.s. *Small* objects. BEVFusion [34] relaxes the limitation on the ego distances and object sizes to some extent by integrating LiDAR point cloud feature with camera image features so that even the distant and small objects contain semantics. Such BEV-based fusion clearly narrows the performance

Table 5: Performance comparisons on nuScenes validation set with object-centric features in different spaces.

#	BEV	Voxel	Image	mAP(%)	NDS (%)
1	✓			68.8	70.9
2		✓		69.1	71.1
3			✓	69.3	71.4
4	✓	✓	✓	**69.8**	**72.3**

Table 6: Performance comparisons on nuScenes validation set with different calibration errors.

Offsets (m)	0.0	0.2	0.4	0.6	0.8	1.0
mAP (%)	69.8	69.7	69.6	69.6	69.5	69.3

gaps by 3.3% and 2.0% when varying ego distances and object sizes. Compared to BEVFusion, our ObjectFusion consistently boosts up the performances under all ego distances and object sizes, and meanwhile further narrowing the performance gaps. The results validate that our object-centric fusion paradigm is more robust against the change of ego distances and object sizes.

Robustness to Calibration Errors. Here we further assess the robustness of ObjectFusion against calibration errors, where the camera and LiDAR are not perfectly aligned. Following TransFusion [1], we randomly add translation offsets to the calibration matrix for evaluation. As shown in Table 6, ObjectFusion demonstrates competitive performances under different offset scales, which surpasses BEVFusion [34] (68.5%) without calibration errors. The results show that the RoI enlarging operation in ObjectFusion is a robust way to compensate for calibration errors.

4.4. Other Experimental Analysis

To evaluate the effectiveness of each component in our method, we conduct ablation studies on the nuScenes validation set. Moreover, we analyze the computational efficiency and generalization ability on more datasets. In the **Supplementary Material**, we provide 1) more ablation studies on voxel sizes, image sizes and data augmentation, 2) robustness to corrupted images, and 3) qualitative results.

Ablation on Object-Centric Features in Different Spaces. We first examine how performance is affected when capitalizing on object-centric features in different spaces. As shown in Table 5, the use of object-centric feature in each BEV/voxel/image space in general achieves a good detection performance. In between, the object-centric BEV feature is inferior to object-centric voxel feature that provides finer geometric information. The object-centric image feature outperforms object-centric voxel feature, showing the advantage of rich texture and semantic information in image feature. Integrating all three kinds of object-centric features in BEV, voxel, and image spaces further boosts up the performances, which demonstrates the complementarity among the three modalities.

Table 7: Performance comparisons on nuScenes validation set with and without (w/o) MSCE.

Method	mAP(%)	NDS (%)
ObjectFusion w/o MSCE	69.3	71.5
ObjectFusion	**69.8**	**72.3**

Table 8: Performance comparisons on Waymo validation set between TransFusion and ObjectFusion.

Method	TransFusion-L [1]	TransFusion [1]	**ObjectFusion**
L2 mAPH(%)	64.9	65.5	66.3

Ablation on the Effect of Modality-Specific Context Encoder (MSCE). Recall that MSCE leverages cross-attention mechanism to contextually encode object-centric features in each space. Table 7 shows the performances of ObjectFusion with and w/o MSCE. In general, the use of MSCE clearly improves the performances of ObjectFusion by 0.5% in mAP, which validates the merit of exploiting inter-object interaction to enhance object-centric features.

Computation Efficiency. Compared to the existing methods (e.g., BEVFusion [34]), the extra computational cost of ObjectFusion is due to the extraction and fusion of object-centric features. Specifically, the inference time of ObjectFusion on an Nvidia V100 GPU is 274ms per sample, which is slightly slower than BEVFusion (257ms). Note that the object-centric features are extracted sequentially in the current implementation. A possible direction for future work is to parallelize this process for faster inference.

Generalization to Waymo Open Dataset. Here we evaluate ObjectFusion on Waymo Open Dataset. Specifically, we followed the setup in TransFusion [1] to train ObjectFusion on Waymo training set and evaluate it on the validation set. Note that the point clouds in Waymo Open Dataset are significantly denser than those in nuScenes, resulting in more accurate detections via LiDAR-only solution and less improvement with multi-modal fusion. As shown in Table 8, ObjectFusion still manages to achieve 0.8% higher LEVEL_2(L2) mAPH than TransFusion.

5. Conclusion

In this work, we circumvent the use of non-trivial inter-modality transformation and propose a new multi-modal fusion paradigm for unifying voxel, BEV, and image features at object level for 3D object detection. To verify our claim, we devise an additional heatmap-based proposal generator to produce 3D object proposals based on BEV features. Such 3D object proposals are further projected into voxel, BEV, and image spaces, yielding spatially aligned object-centric features in each modality. All three object-centric features are finally unified at object level for detection. We empirically validate the superiority of our object-centric fusion paradigm over the state-of-the-art approaches for multi-modal 3D object detection.

References

[1] Xuyang Bai, Zeyu Hu, Xinge Zhu, Qingqiu Huang, Yilun Chen, Hongbo Fu, and Chiew-Lan Tai. Transfusion: Robust lidar-camera fusion for 3d object detection with transformers. In *CVPR*, 2022. 4, 5, 6, 7, 8

[2] Alex Bewley, Pei Sun, Thomas Mensink, Dragomir Anguelov, and Cristian Sminchisescu. Range conditioned dilated convolutions for scale invariant 3d object detection. In *CoRL*, 2020. 2

[3] Garrick Brazil and Xiaoming Liu. M3d-rpn: Monocular 3d region proposal network for object detection. In *ICCV*, 2019. 2

[4] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020. 5, 6

[5] Qi Cai, Yingwei Pan, Chong-Wah Ngo, Xinmei Tian, Lingyu Duan, and Ting Yao. Exploring object relation in mean teacher for cross-domain detection. In *CVPR*, 2019. 2

[6] Qi Cai, Yingwei Pan, Yu Wang, Jingen Liu, Ting Yao, and Tao Mei. Learning a unified sample weighting network for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14173–14182, 2020. 2

[7] Q Cai, Y Pan, T Yao, and T Mei. 3d cascade rcnn: High quality object detection in point clouds. *IEEE Transactions on Image Processing: a Publication of the IEEE Signal Processing Society*, 31:5706–5719, 2022. 2

[8] Yuning Chai, Pei Sun, Jiquan Ngiam, Weiyue Wang, Benjamin Caine, Vijay Vasudevan, Xiao Zhang, and Dragomir Anguelov. To the point: Efficient 3d object detection in the range image with graph convolution kernels. In *CVPR*, 2021. 2

[9] Qi Chen, Lin Sun, Zhixin Wang, Kui Jia, and Alan Yuille. Object as hotspots: An anchor-free 3d object detection approach via firing of hotspots. In *ECCV*, 2020. 2

[10] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *CVPR*, 2017. 2

[11] MMDetection3D Contributors. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. https://github.com/open-mmlab/mmdetection3d, 2020. 6

[12] Jiajun Deng, Shaoshuai Shi, Pei-Cian Li, Wen gang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. In *AAAI*, 2021. 2, 4

[13] Lue Fan, Ziqi Pang, Tianyuan Zhang, Yu-Xiong Wang, Hang Zhao, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Embracing single stride 3d object detector with sparse transformer. In *CVPR*, 2022. 2

[14] Lue Fan, Xuan Xiong, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Rangedet: In defense of range view for lidar-based 3d object detection. In *ICCV*, 2021. 2

[15] Runzhou Ge, Zhuangzhuang Ding, Yihan Hu, Wenxin Shao, Li Huang, Kun Li, and Qiang Liu. Real-time anchor-free single-stage 3d detection with iou-awareness. *arXiv preprint arXiv:2107.14342*, 2021. 2

[16] Ross Girshick. Fast r-cnn. In *CVPR*, pages 1440–1448, 2015. 2

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 2, 4, 6

[18] Junjie Huang, Guan Huang, Zheng Zhu, and Dalong Du. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. *arXiv preprint arXiv:2112.11790*, 2021. 2, 7

[19] Tengteng Huang, Zhe Liu, Xiwu Chen, and Xiang Bai. Epnet: Enhancing point features with image semantics for 3d object detection. In *ECCV*, 2020. 3

[20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICML*, 2014. 6

[21] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *IROS*, 2018. 2

[22] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *CVPR*, 2019. 2, 6

[23] Yanwei Li, Yilun Chen, Xiaojuan Qi, Zeming Li, Jian Sun, and Jiaya Jia. Unifying voxel-based representation with transformer for 3d object detection. In *NeurIPS*, 2022. 2

[24] Yinhao Li, Zheng Ge, Guanyi Yu, Jinrong Yang, Zengran Wang, Yukang Shi, Jianjian Sun, and Zeming Li. Bevdepth: Acquisition of reliable depth for multi-view 3d object detection. *arXiv preprint arXiv:2206.10092*, 2022. 2

[25] Yehao Li, Yingwei Pan, Ting Yao, and Tao Mei. Comprehending and ordering semantics for image captioning. In *CVPR*, pages 17990–17999, 2022. 5

[26] Yanwei Li, Xiaojuan Qi, Yukang Chen, Liwei Wang, Zeming Li, Jian Sun, and Jiaya Jia. Voxel field fusion for 3d object detection. In *CVPR*, 2022. 3

[27] Yehao Li, Ting Yao, Yingwei Pan, and Tao Mei. Contextual transformer networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):1489–1500, 2022. 5

[28] Zhichao Li, Feng Wang, and Naiyan Wang. Lidar r-cnn: An efficient and universal 3d object detector. In *CVPR*, 2021. 2

[29] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Qiao Yu, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *ECCV*, 2022. 2

[30] Tingting Liang, Hongwei Xie, Kaicheng Yu, Zhongyu Xia, Zhiwei Lin, Yongtao Wang, Tao Tang, Bing Wang, and Zhi Tang. Bevfusion: A simple and robust lidar-camera fusion framework. In *NeurIPS*, 2022. 1, 2, 3, 6

[31] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 3

[32] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petr: Position embedding transformation for multi-view 3d object detection. In *ECCV*, 2022. 2

[33] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 3, 6

[34] Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela Rus, and Song Han. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. *arXiv preprint arXiv:2205.13542*, 2022. 1, 2, 3, 4, 5, 6, 7, 8

[35] Gregory P Meyer, Ankit Laddha, Eric Kee, Carlos Vallespi-Gonzalez, and Carl K Wellington. Lasernet: An efficient probabilistic 3d object detector for autonomous driving. In *CVPR*, 2019. 2

[36] Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3d object detection. In *ICCV*, 2021. 2

[37] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. In *CVPR*, 2017. 2

[38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 6

[39] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *ECCV*, 2020. 2

[40] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *CVPR*, 2018. 2

[41] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 2

[42] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 2

[43] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *CVPR*, 2020. 2

[44] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *CVPR*, 2019. 2

[45] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *ICCV*, 2019. 2

[46] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, 2019. 2

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2, 5

[48] Sourabh Vora, Alex H Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential fusion for 3d object detection. In *CVPR*, 2020. 1, 3, 6

[49] Chunwei Wang, Chao Ma, Ming Zhu, and Xiaokang Yang. Pointaugmenting: Cross-modal augmentation for 3d object detection. In *CVPR*, 2021. 1, 3, 6

[50] Tai Wang, ZHU Xinge, Jiangmiao Pang, and Dahua Lin. Probabilistic and geometric depth: Detecting objects in perspective. In *CoRL*, 2022. 2

[51] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Fcos3d: Fully convolutional one-stage monocular 3d object detection. In *ICCV*, 2021. 2

[52] Yue Wang, Vitor Campagnolo Guizilini, Tianyuan Zhang, Yilun Wang, Hang Zhao, and Justin Solomon. Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In *CoRL*, 2022. 2

[53] Xiaopei Wu, Liang Peng, Honghui Yang, Liang Xie, Chenxi Huang, Chengqi Deng, Haifeng Liu, and Deng Cai. Sparse fuse dense: Towards high quality 3d detection with depth completion. In *CVPR*, 2022. 3

[54] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3d object detection from monocular images. In *CVPR*, 2018. 2

[55] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 2018. 2, 6

[56] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *CVPR*, 2020. 2

[57] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Std: Sparse-to-dense 3d object detector for point cloud. In *ICCV*, 2019. 2

[58] Ting Yao, Yehao Li, Yingwei Pan, Yu Wang, Xiao-Ping Zhang, and Tao Mei. Dual vision transformer. *IEEE transactions on pattern analysis and machine intelligence*, 2023. 5

[59] Ting Yao, Yingwei Pan, Yehao Li, Chong-Wah Ngo, and Tao Mei. Wave-vit: Unifying wavelet and transformers for visual representation learning. In *ECCV*, pages 328–345. Springer, 2022. 5

[60] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *CVPR*, 2021. 2, 6, 7

[61] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Multi-modal virtual point 3d detection. In *NeurIPS*, 2021. 3, 6

[62] Yanan Zhang, Jiaxin Chen, and Di Huang. Cat-det: Contrastively augmented transformer for multi-modal 3d object detection. In *CVPR*, 2022. 3

[63] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, 2018. 2, 3, 6

[64] Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. Class-balanced grouping and sampling for point cloud 3d object detection. *arXiv preprint arXiv:1908.09492*, 2019. 6