

PolarFormer: Multi-camera 3D Object Detection with Polar Transformers

Yanqin Jiang^{1,2} Li Zhang^{1*} Zhenwei Miao³ Xiatian Zhu⁴
Jin Gao² Weiming Hu² Yu-Gang Jiang¹

¹Fudan University ²CASIA ³Alibaba DAMO Academy ⁴University of Surrey

<https://github.com/fudan-zvg/PolarFormer>

Abstract

3D object detection in autonomous driving aims to reason “what” and “where” the objects of interest present in a 3D world. Following the conventional wisdom of previous 2D object detection, existing 3D object detection methods often adopt the canonical Cartesian coordinate system with perpendicular axis. However, we conjugate that this does not fit the nature of the ego car’s perspective, as each onboard camera perceives the world in shape of wedge intrinsic to the imaging geometry with radical (non-perpendicular) axis. Hence, in this paper we advocate the exploitation of the Polar coordinate system and propose a new Polar Transformer (**PolarFormer**) for more accurate 3D object detection in the bird’s-eye-view (BEV) taking as input only multi-camera 2D images. Specifically, we design a cross-attention based Polar detection head without restriction to the shape of input structure to deal with irregular Polar grids. For tackling the unconstrained object scale variations along Polar’s distance dimension, we further introduce a multi-scale Polar representation learning strategy. As a result, our model can make best use of the Polar representation rasterized via attending to the corresponding image observation in a sequence-to-sequence fashion subject to the geometric constraints. Thorough experiments on the nuScenes dataset demonstrate that our PolarFormer outperforms significantly state-of-the-art 3D object detection alternatives, as well as yielding competitive performance on BEV semantic segmentation task.

1 Introduction

3D object detection is an enabling capability of autonomous driving in unconstrained real-world scenes [37, 35]. It aims to predict the location, dimension and orientation of individual objects of interest in a 3D world. Despite favourable cost advantages, multi-camera based 3D object detection [35, 34, 37, 44, 19] remains particularly challenging. To obtain 3D representation, dense depth estimation is often leveraged [23], which however is not only expensive in computation but error prone. To bypass depth inference, more recent methods [37, 16] exploit query-based 2D detection [4] to learn a set of virtual embedding in Cartesian coordinate frame for multi-camera 3D object detection. Typically, the canonical Cartesian coordinate system with *perpendicular* axis is adopted in either 2D [44, 35] or 3D [37, 16] space.

In contrast, the physical world perceived under each camera *in the ego car’s perspective* is in shape of wedge intrinsic to the camera imaging geometry with radical *non-perpendicular* axis (Figure 5). Bearing this imaging property in mind, we conjugate that the Polar coordinate system should be more appropriate and natural than the often adopted Cartesian counterpart for 3D object detection. For example, those objects near the ego vehicle would be allowed to involve richer representations

*Li Zhang (lizhangfd@fudan.edu.cn) is the corresponding author with School of Data Science, Fudan University.

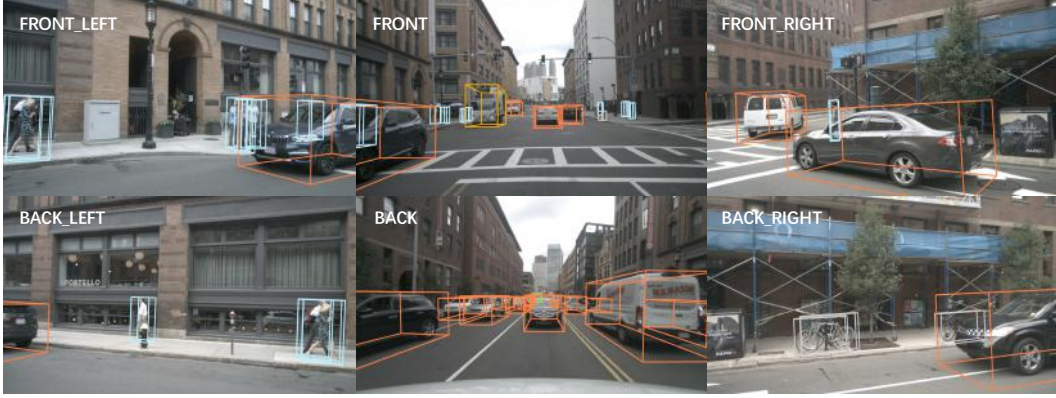


Figure 1: Taking multi-camera images as input, the proposed PolarFormer model is designed particularly for accurate 3D object detection in the Polar coordinate system.

than distant ones in Polar coordinate. In the Cartesian coordinate, however, the near/distant objects have to be down-sampled/up-sampled proportionally. In particular, such object down-sampling could result in information loss whilst up-sampling is typically with little effect on information enhancement (Figure 5). Indeed, the Polar coordinate has been exploited in a few LiDAR-based 3D perception methods [43, 1, 24, 45]. However, they are limited algorithmically due to the adoption of convolutional networks restricted to rectangular grid structure and local receptive fields.

Motivated by the aforementioned insights, in this work a novel *Polar Transformer* (PolarFormer) model for multi-camera 3D object detection in a Polar coordinate system is introduced (Figure 3). Specifically, we first learn the representation of polar rays corresponding to image regions in a sequence-to-sequence cross-attention formulation. Then we rasterize a Polar BEV representation consisting of a set of Polar rays evenly distributed around 360 degrees. To deal with irregular Polar grids as suffered by conventional LiDAR based solutions [43, 1, 24, 45], we propose a cross-attention based decoder head design without restriction to the shape of input structure. For tackling the challenge of unconstrained object scale variation along Polar’s distance dimension, we resort to a multi-scale Polar BEV representation learning strategy.

The **contributions** of this work are summarized as follows: **(I)** In this paper, we conjugate that the Polar coordinate system is more appropriate and natural than the often adopted Cartesian counterpart for 3D object detection and propose a new *Polar Transformer* (PolarFormer) model for multi-camera 3D object detection in the Polar coordinate system; **(II)** This is achieved based on two Polar-tailored designs: A cross-attention based decoder design for dealing with the irregular Polar grids, and a multi-scale Polar representation learning strategy for handling the unconstrained object scale variations over Polar’s distance dimension; **(III)** Extensive experiments on the nuScenes dataset show that our PolarFormer establishes new state of the art for camera-based 3D object detection (Figure 1), whilst yielding top performance on the BEV semantic segmentation.

2 Related work

Monocular/multi-camera 3D object detection Image-based 3D object detection aims to estimate the object location, dimension and orientation in the 3D space alongside its category given only image input. To solve this ill-posed problem, [44, 35] naively build their detection pipelines by augmenting 2D detectors [44, 30] in a data driven fashion. PGD [34] further captures the uncertainty and models the relationship of different objects by utilizing geometric prior. Contrast to the above image-only methods, depth-based methods [39, 8, 36, 42, 20, 25, 32, 33] use depth cues as 3D information to mitigate the naturally ill-posed problem. Recently, multi-camera-based 3D object detection emerges. DETR3D [37] considers detecting objects across all cameras collectively. It learns a set of sparse and virtual query embedding, without explicitly building the geometry structure among objects/queries. [16] considers detecting objects in BEV, performing end-to-end object detection via object queries. Note that multi-camera setting uses the same amount of training data as the monocular pipelines. Both multi-camera and monocular paradigms share the same evaluation metrics.

Bird’s-eye-view (BEV) representation Recently there is a surge of interest in transforming the monocular or multi-view images from ego car cameras into the bird’s-eye-view coordinate [27, 23, 15, 26, 25, 28] followed by specific optimization tasks (e.g., 3D object detection, semantic segmentation). A natural solution [23, 12] is to learn the BEV representation by leveraging the pixel-level dense depth estimation. This however is error-prone due to lacking ground-truth supervision. Another line of research aims to bypass the depth prediction and directly leverage a Transformer [6, 3, 28, 19] or a FC layer [15, 26, 40] to learn the transformation from camera inputs to the BEV coordinate. A similar attempt as ours is conducted in [28] but limited in a couple of aspects: (i) It is restricted to monocular input for a straightforward 2D segmentation task while we consider multiple cameras collectively for more challenging 3D object detection; (ii) It further resamples the Polar BEV map into the Cartesian coordinate for the downstream segmentation prediction as the latter is required by the convolution-based decoder, whilst our method learns the Polar BEV feature representation and predictions in a unified Polar coordinate system; (iii) We uniquely provide a solid multi-scale Polar BEV transformation to tackle the unconstrained object scale variations and followed by a jointly optimized cross-attention based Polar BEV decoder.

3D object detection in Polar coordinate 3D object detection in the Polar or Polar-like coordinate system has been attempted in LiDAR-based perception methods. For example, CyliNet [45] introduces range-based guidance for extracting Polar-consistent features. In particular, it adapts a Cartesian heatmap to a Polar version for object classification, whilst learning relative heading angles and velocities. However, CyliNet still lags clearly behind the Cartesian counterpart. Recently, PolarSteam [5] designs a learnable sampling module for relieving object distortion in Polar coordinate and uses range-stratified convolution and normalization for flexibly extracting the features over different ranges. Limited by the convolution based network, it remains inferior despite of these special designs. In contrast to all these works, we resort to the cross-attention mechanism, tackling the challenges of object scale variance and appearance distortion in the Polar coordinate principally.

3 Method

In 3D object detection task, we are given a set of N monocular views $\{\mathbf{I}_n, \mathbf{\Pi}_n, \mathbf{E}_n\}_{n=1}^N$ consisting of input images $\mathbf{I}_n \in \mathbb{R}^{H \times W \times 3}$, camera intrinsics $\mathbf{\Pi}_n \in \mathbb{R}^{3 \times 3}$ and camera extrinsics $\mathbf{E}_n \in \mathbb{R}^{4 \times 4}$. The objective of our *Polar Transformer* (PolarFormer) is to learn an effective Polar BEV representation from multiple camera views for facilitating the prediction of object locations, dimensions, orientations and velocities in the Polar coordinate system. PolarFormer consists of the following components. A *cross-plane encoder* first produces a multi-scale feature representation of each input image, characterized by a cross-plane attention mechanism in which Polar queries attend to input images to generate 3D features in BEV. A *Polar alignment module* then aggregates Polar rays from multiple camera views to generate a structured Polar map. Further, a *Polar BEV encoder* enhances the Polar features with multi-scale feature interaction. Finally, a *Polar detection head* decodes the Polar map and predicts the objects in the Polar coordinate system. For tackling the unconstrained object scale variation with multi-granularity of details, we consider a multi-scale Polar BEV representation structure. As shown in Figure 4, image features with different scales have unique cross-plane encoders and interact with each other in a shared Polar BEV encoder. Multi-scale Polar BEV maps are then queried by Polar BEV decoder head. An overall architecture of PolarFormer is depicted in Figure 2.

3.1 Cross-plane encoder

The goal of cross-plane encoder is to associate an image with Polar rays. According to the geometric model of camera, for any camera coordinate $(x^{(C)}, y^{(C)}, z^{(C)}) \in \mathbb{R}^3$, the transformation to image coordinate $(x^{(I)}, y^{(I)}) \in \mathbb{R}^2$ could be described as:

$$s \begin{bmatrix} x^{(I)} \\ y^{(I)} \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^{(C)} \\ y^{(C)} \\ z^{(C)} \end{bmatrix}, \quad (1)$$

where f_x, f_y, u_0 and v_0 are camera intrinsic parameters in $\mathbf{\Pi}$, $x^{(C)}, y^{(C)}$ and $z^{(C)}$ are the horizontal, vertical, depth coordinate respectively. s is the scale factor. For any Polar BEV coordinate

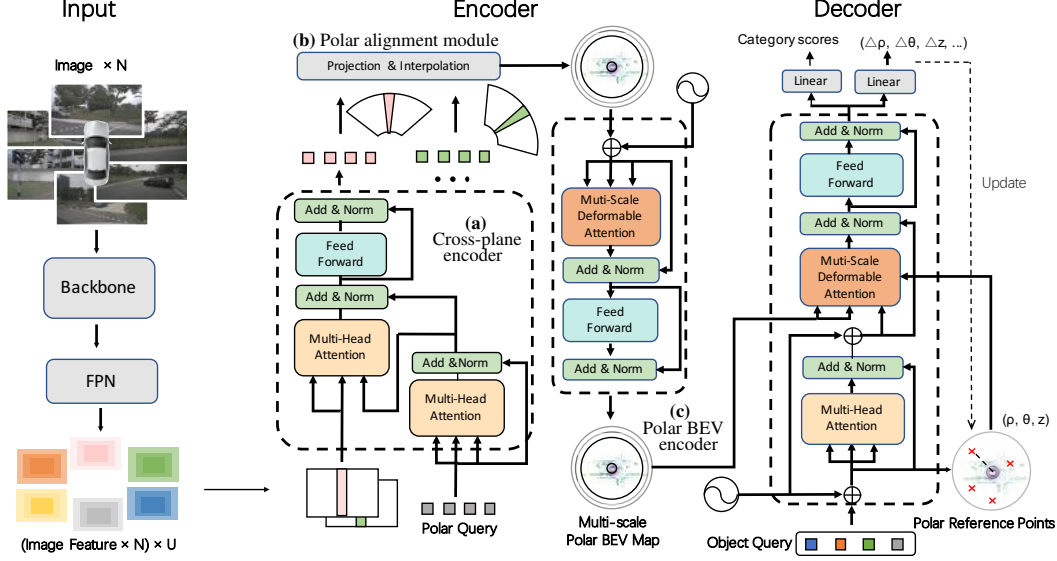


Figure 2: Schematic illustration of our proposed *PolarFormer* for multi-camera 3D object detection. For each image captured by any camera view, our model first extracts the feature maps at multiple spatial scales. Given such a feature map, the cross-plane encoder (a) then transforms all the feature columns to a set of Polar rays in a sequence-to-sequence manner via polar queries based cross-attention. The polar rays from all the cameras are subsequently processed by a Polar alignment module (b) to generate a structured multi-scale Polar BEV map, followed by further enhancement via interactions among different scales using a Polar BEV encoder (c). At last, a specially designed Polar Head decodes multi-scale Polar BEV features for making final predictions in the Polar coordinate.

$(\rho^{(P)}, \phi^{(P)})$, we have:

$$\phi^{(P)} = \arctan \frac{x^{(C)}}{z^{(C)}} = \arctan \frac{x^{(I)} - u_0}{f_x}, \quad (2)$$

$$\rho^{(P)} = \sqrt{(x^{(C)})^2 + (z^{(C)})^2} = z^{(C)} \sqrt{\left(\frac{x^{(I)} - u_0}{f_x}\right)^2 + 1}. \quad (3)$$

Eq. (2) suggests that the azimuth $\phi^{(P)}$ is irrelevant to the vertical value of image coordinate. It is hence natural to build a one-to-one relationship between Polar rays and image columns [28]. However, we need object depth $z^{(C)}$ to compute radius $\rho^{(P)}$, the estimation of which is ill-posed. Instead of explicit depth estimation, we leverage the attention mechanism [31] to model the relationship between pixels along the image column and positions along the Polar ray.

Let $\mathbf{f}_{n,u,w} \in \mathbb{R}^{H_u \times C}$ represent the image column from n th camera, u th scale and w th column, and $\dot{\mathbf{p}}_{n,u,w} \in \mathbb{R}^{R_u \times C}$ denote the corresponding *Polar ray* query we introduce, where H and R are the image feature map's height and Polar map's range. We formulate cross-plane attention as:

$$\mathbf{p}_{n,u,w} = \text{MultiHead}(\dot{\mathbf{p}}_{n,u,w}, \mathbf{f}_{n,u,w}, \mathbf{f}_{n,u,w}) = \text{Concat}(\text{head}_1, \dots, \text{head}_h) \mathbf{W}_u^O, \quad (4)$$

where $\text{head}_i = \text{Attention}(\dot{\mathbf{p}}_{n,u,w} \mathbf{W}_{i,u}^Q, \mathbf{f}_{n,u,w} \mathbf{W}_{i,u}^K, \mathbf{f}_{n,u,w} \mathbf{W}_{i,u}^V),$

where $\mathbf{W}_{i,u}^Q \in \mathbb{R}^{d_{\text{model}} \times d_q}$, $\mathbf{W}_{i,u}^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $\mathbf{W}_{i,u}^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$, $\mathbf{W}_u^O \in \mathbb{R}^{h d_{\text{model}} \times d_k}$ are the projection parameters, $d_q = d_k = d_v = d_{\text{model}}/h$, d is the feature dimension and h is the number of heads.

Stacking the Polar ray features $\mathbf{p}_{n,u,w} \in \mathbb{R}^{R_u \times C}$ along azimuth axis, we obtain the Polar feature map (i.e., *Polar BEV representation*) $\mathbf{P}_{n,u}$ for n th camera and u th scale as:

$$\mathbf{P}_{n,u} = \text{Stack}([\mathbf{p}_{n,u,1}, \dots, \mathbf{p}_{n,u,W_u}], \text{dim} = 1) \in \mathbb{R}^{R_u \times W_u \times C}, \quad (5)$$

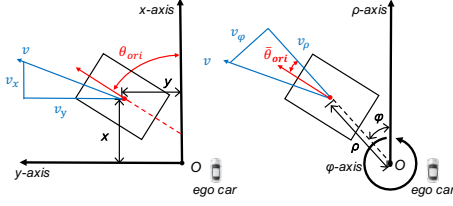


Figure 3: Cartesian and Polar coordinates.

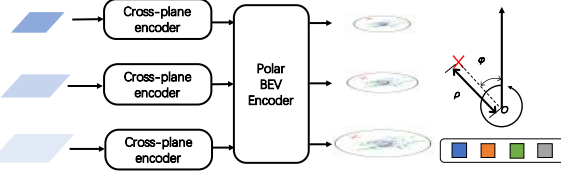


Figure 4: Multi-scale Polar BEV maps.

where W_u denotes the azimuth dimension. This sequence-to-sequence cross-attention-based encoder can encode geometric imaging prior and implicitly learn a proxy for depth efficiently. Next, we show how to integrate independent Polar rays from multiple cameras into a coherent and structured Polar BEV map.

3.2 Polar alignment across multiple cameras

Our Polar alignment module transforms Polar rays from different camera coordinates to a shared world coordinate. Taking multi-view Polar feature maps $\{\mathbf{P}_{n,u}\}_{n=1}^N$ and camera matrix $\{\mathbf{\Pi}_n, \mathbf{E}_n\}_{n=1}^N$ as inputs, it produces a coherent Polar BEV map $\mathbf{G}_u \in \mathbb{R}^{\mathcal{R}_u \times \mathcal{N}_u \times \mathcal{C}}$, covering all camera views, where $\mathcal{R}_u, \mathcal{N}_u$ and \mathcal{C} are the dimensions of radius, azimuth and feature. Concretely, it first generates a set of 3D points in the cylindrical coordinate uniformly, denoted by $\mathcal{G}^{(P)} = \{(\rho_i^{(P)}, \phi_j^{(P)}, z_k^{(P)}) | i = 1, \dots, \mathcal{R}_u; j = 1, \dots, \mathcal{N}_u; k = 1, \dots, \mathcal{Z}_u\}$, where \mathcal{Z}_u is the number of points along z axis. Since cylindrical coordinate and Polar coordinate share radius and azimuth axis, their superscripts are both denoted with P . The points are then projected to the image plane of n th camera to retrieve the index of Polar ray, estimated by:

$$\begin{bmatrix} sx_{i,j,k,n}^{(I)} \\ sy_{i,j,k,n}^{(I)} \\ s \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{\Pi}_n & 0 \\ 0 & 1 \end{bmatrix} \mathbf{E}_n \begin{bmatrix} \rho_i^{(P)} \sin \phi_j^{(P)} \\ \rho_i^{(P)} \cos \phi_j^{(P)} \\ z_k^{(P)} \\ 1 \end{bmatrix}, \quad (6)$$

where s is the scale factor. Coherent Polar BEV map of u th scale can be then generated by:

$$\begin{aligned} \mathbf{G}_u(\rho_i^{(P)}, \phi_j^{(P)}) &= \frac{1}{\sum_{n=1}^N \sum_{k=1}^{\mathcal{Z}} \lambda_n(\rho_i^{(P)}, \phi_j^{(P)}, z_k^{(P)})} \\ &\cdot \sum_{n=1}^N \sum_{k=1}^{\mathcal{Z}} \lambda_n(\rho_i^{(P)}, \phi_j^{(P)}, z_k^{(P)}) \mathcal{B}(\mathbf{P}_{n,u}, (\bar{x}_{i,j,k,n}^{(I)}, \bar{r}_{i,j,n})), \end{aligned} \quad (7)$$

where $\lambda_n(\rho_i^{(P)}, \phi_j^{(P)}, z_k^{(P)})$ is binary weighted factor indicating visibility in n th camera, $\mathcal{B}(\mathbf{P}, (x, y))$ denotes the bilinear sampling \mathbf{P} at location (x, y) , $\bar{x}_{i,j,k,n}^{(I)}$ and $\bar{r}_{i,j,n}$ denote the normalized Polar ray index and radius index. Note the radius r is the distance between the point and the camera origin in BEV. Our Polar alignment module incorporates the features at different heights by generating the points along z axis. As validated in Table 3a, learning Polar representation is superior over Cartesian coordinate due to minimal information loss and higher consistency with raw visual data.

3.3 Polar BEV encoder at multiple scales

We leverage multi-scale feature maps for handling object scale variance in the Polar coordinate. To that end, the Polar BEV encoder performs information exchange among neighbouring pixels and across multi-scale feature maps. Formally, let $\{\mathbf{G}_u\}_{u=1}^U$ be the input multi-scale Polar feature maps and $\hat{x}_q \in [0, 1]^2$ be the normalized coordinates of the reference points for each query element q , we introduce a multi-scale deformable attention module [46] as:

$$\text{MSDeformAttn}(\mathbf{z}_q, x_q, \{\mathbf{G}_u\}_{u=1}^U) = \sum_{m=1}^M \mathbf{W}_m \left[\sum_{u=1}^U \sum_{k=1}^K A_{muqk} \mathbf{W}'_m \mathbf{G}_u(\zeta_u(\hat{x}_q) + \Delta x_{muqk}) \right], \quad (8)$$

where m and k are the index of the attention head and the sampling point. \mathbf{z}_q is the query feature. Δx_{muqk} and A_{muqk} denote the sampling offset and the attention weight of the k th sampling point in u th feature level and m th attention head. The attention weight A_{muqk} is normalized by $\sum_{u=1}^U \sum_{k=1}^K A_{muqk} = 1$. Function ζ_u generates the sampling offsets and rescales the normalized coordinate \hat{x}_q to the u th feature scale. \mathbf{W}_m and \mathbf{W}'_m are learnable parameters. Serving as query, each pixel in the multi-scale feature maps exploits the information from both neighbouring pixels and pixels across scales. This enables learning richer semantics across all feature scales.

3.4 Polar BEV decoder at multiple scales

The Polar BEV decoder decodes the above multi-scale Polar features to make predictions in the Polar coordinate. We construct the Polar BEV decoder with deformable attention [46]. Specifically, we query q in Eq. (8) as learnable parameters.

Unlike 2D reference points in the encoder, here the reference points are in 3D cylindrical coordinate, equal to Polar coordinate when projected to BEV. The classification branch in each decoder layer outputs the confidence score vector $\mathbf{c} \in \mathbb{R}^{\mathcal{O}}$, where \mathcal{O} is the number of categories. The key learning targets of regression branch are in polar coordinate instead of Cartesian coordinate, as illustrated in Figure 3. For simplicity, superscript (P) is omitted. Reference points (ρ, ϕ, z) are iteratively refined in the decoder. With reference points, the regression branch regresses the offsets d_ρ, d_ϕ and d_z . The learning targets for orientation θ and velocity v are relative to azimuths of objects and separated to orthogonal components $\theta_\phi, \theta_\rho, v_\phi$ and v_ρ , defined by:

$$\bar{\theta}_{ori} = \theta_{ori} - \phi, \quad \theta_\phi = \sin \bar{\theta}_{ori}, \quad \theta_\rho = \cos \bar{\theta}_{ori}, \quad (9)$$

and

$$\bar{\theta}_v = \theta_v - \phi, \quad v_\phi = v_{abs} \sin \bar{\theta}_v, \quad v_\rho = v_{abs} \cos \bar{\theta}_v. \quad (10)$$

Here, θ_{ori} is the yaw angle of the bounding box. v_{abs} and θ_v are the absolute value and angle of velocity. We regress the object size l, w and h as $\log l, \log w$ and $\log h$. We adopt Focal loss [17] and L1 loss for classification and regression respectively.

4 Experiments

Dataset We evaluate the PolarFormer on the nuScenes dataset [2]. It provides images with a resolution of 1600×900 from 6 surrounding cameras (Figure 1). The total of 1000 scenes, where each sequence is roughly 20 seconds long and annotated every 0.5 second, is split officially into train/val/test set with 700/150/150 scenes. It comes with informative annotations of the map. We choose five classes (drivable area, pedestrian crossing, walkway, carpark, and divider) for map semantic segmentation.

Metrics For 3D object detection, we follow the default evaluation protocol [2]. There are 7 official metrics: mean Average Precision (mAP), Average Translation Error (ATE), Average Scale Error (ASE), Average Orientation Error (AOE), Average Velocity Error (AVE), Average Attribute Error (AAE), and NuScenes Detection Score (NDS). Particularly, NDS is the composite of the other indicators for comprehensive judging.

Implementation details We implement our approach based on the codebase mmdetection3d [7]. Following DETR3D [37] and FCOS3D [35], a ResNet-101 [11], with 3rd and 4th stages equipped with deformable convolutions, is adopted as the backbone architecture. The number of cross-plane encoder layer is set to 3 for each feature scale. The resolution of radius and azimuth for our multi-scale Polar BEV maps are (64, 256), (32, 128), (16, 64) respectively. We use 6 Polar BEV encoder and 6 decoder layers. Following DETR3D [37], our backbone is initialized from a checkpoint of FCOS3D [35] trained on nuScenes 3D detection task, while the rest is initialized randomly. We use the above setting for prototype verification. To fully leverage the sequence data, we further conduct temporal fusion between the current frame and one history sweep in the BEV space. Following BEVDet4D [13], we simply concatenate two temporally adjacent multi-scale Polar BEV maps along the feature dimension and feed it to the Polar BEV encoder. We randomly sample a history sweep from [3T; 27T] during training, and sample the frame at 15T for inference. T ($\approx 0.083s$) refers to the time interval between two sweep frames. We term our temporal version as PolarFormer-T.

Table 1: State-of-the-art comparison on nuScenes test set. † denotes the prototype setting: The model is initialized from a FCOS3D [35] checkpoint trained on the nuScenes 3D detection dataset. ‡ denotes the improved setting: A pretrained model from DD3D [22] is used, which includes external data from DDAD [10]. * denotes backbone is pretrained on COCO [18] and nuImage [2].

Methods	Backbone	mAP↑	NDS↑	mATE↓	mASE↓	mAOE↓	mAVE↓	mAAE↓
FCOS3D [†] [35]	R101	35.8	42.8	69.0	24.9	45.2	143.4	12.4
PGD [†] [34]	R101	38.6	44.8	62.6	24.5	45.1	150.9	12.7
Ego3RT [†] [19]	R101	38.9	44.3	59.9	26.8	47.0	116.9	17.2
BEVFormer-S [†] [16]	R101	40.9	46.2	65.0	26.1	43.9	92.5	14.7
PolarFormer [†]	R101	41.5	47.0	65.7	26.3	40.5	91.1	13.9
BEVFormer [†] [16]	R101	44.5	53.5	63.1	25.7	40.5	43.5	14.3
PolarFormer-T [†]	R101	45.7	54.3	61.2	25.7	39.2	46.7	12.9
DETR3D [‡] [37]	V2-99	41.2	47.9	64.1	25.5	39.4	84.5	13.3
M2BEV* [38]	X101	42.9	47.4	58.3	25.4	37.6	10.53	19.0
Ego3RT [‡] [19]	V2-99	42.5	47.9	54.9	26.4	43.3	101.4	14.5
BEVFormer-S [‡]	V2-99	43.5	49.5	58.9	25.4	40.2	84.2	13.1
PolarFormer [‡]	V2-99	45.5	50.3	59.2	25.8	38.9	87.0	13.2
BEVFormer [‡] [16]	V2-99	48.1	56.9	58.2	25.6	37.5	37.8	12.6
PolarFormer-T [‡]	V2-99	49.3	57.2	55.6	25.6	36.4	44.0	12.7

Training Following DETR3D [37] we train our models for 24 epochs with the AdamW optimizer and cosine annealing learning rate scheduler on 8 NVIDIA V100 GPUs. The initial learning rate is 2×10^{-4} , and the weight decay is set to 0.075. Total batch size is set to 48 across six cameras. Synchronized batch normalization is adopted. All experiments use the original input resolution. Note our image variant uses the same amount of training data as the monocular pipelines [35] and the multi-camera counterparts [37, 16]. Multi-camera and monocular paradigms share the same evaluation metrics.

Inference We evaluate our model on both nuScenes validation set and test server. We do not adopt model-agnostic tricks such as model ensemble and test time augmentation.

4.1 Comparison with the state of the art

We compare our method with the state of the art on both val and test sets of nuScenes. In addition to the (i) prototype setting mentioned in implementation details, we also evaluate our model in the (ii) improved setting, with VoVNet (V2-99) [14] as backbone architecture with a pretrained checkpoint from DD3D [22] (fine-tuned on extra DDAD15M [10] dataset) to boost performance.

Table 1 compares the results on nuScenes test set. We observe that our PolarFormer achieves the best performance under both the (i) prototype and (ii) improved setting in terms of mAP and NDS metrics. With temporal information PolarFormer-T can further boost performance substantially.

Table 2 shows that our method achieves leading performance on the val set for both mAP and NDS. Under the improved setting, PolarFormer shines on all metrics except mASE and mAAE. Again, PolarFormer-T beats the alternative BEVFormer by a clear margin, indicating the superiority of learning representation in the Polar coordinate. More qualitative results are shown in supplementary materials.

4.2 Ablation studies

We conduct a series of ablation studies on nuScenes val set to validate the design of PolarFormer. Each proposed component and important hyperparameters are examined thoroughly.

Polar v.s. Cartesian Table 3a ablates the coordinate system. We make several observations: (I) Learning the representation and making the prediction both on Cartesian, Centerpoint [41] gives a strong baseline with 0.378 mAP and 0.454 NDS; After applying circle NMS, Centerpoint* can

Table 2: State-of-the-art comparison on nuScenes val set. † denotes the prototype setting: The model is initialized from a FCOS3D [35] checkpoint trained on the nuScenes 3D detection dataset. ‡ denotes improved setting: A pretrained model from DD3D [22] is used, which includes extra data from DDAD [10].* denotes backbone is pretrained on COCO [18] and nuImage [2].

Methods	Backbone	mAP↑	NDS↑	mATE↓	mASE↓	mAOE↓	mAVE↓	mAAE↓
FCOS3D [†] [35]	R101	32.1	39.5	75.4	26.0	48.6	133.1	15.8
DETR3D [†] [37]	R101	34.7	42.2	76.5	26.7	39.2	87.6	21.1
PGD [†] [34]	R101	35.8	42.5	66.7	26.4	43.5	127.6	17.7
Ego3RT [†] [19]	R101	37.5	45.0	65.7	26.8	39.1	85.0	20.6
BEVFormer-S [†] [16]	R101	37.5	44.8	72.5	27.2	39.1	80.2	20.0
PolarFormer [†]	R101	39.6	45.8	70.0	26.9	37.5	83.9	24.5
Ego3RT [‡] [19]	V2-99	47.8	53.4	58.2	27.2	31.6	68.3	20.2
M2BEV* [38]	X101	41.7	47.0	64.7	27.5	37.7	83.4	24.5
PolarFormer [‡]	V2-99	50.0	56.2	58.3	26.2	24.7	60.1	19.3
BEVFormer [†] [16]	R101	41.6	51.7	67.3	27.4	37.2	39.4	19.8
PolarFormer-T [†]	R101	43.2	52.8	64.8	27.0	34.8	40.9	20.1

Table 3: 3D object detection results in different coordinate systems and ablations for the model architecture. PC denotes feature in Polar and prediction in Cartesian.

Method	Feature	Prediction	mAP↑	NDS↑	Position encoding	mAP↑	NDS↑
Centerpoint [41]	Cartesian	Cartesian	37.8	45.4	2D learnable PE	38.8	45.1
Centerpoint* [41]	Cartesian	Cartesian	38.5	45.6	3D PE	38.5	44.9
PolarFormer-CC	Cartesian	Cartesian	38.1	45.5	Fixed Sine PE	39.6	45.8
PolarFormer-PC	Polar	Cartesian	38.5	45.0			
PolarFormer	Polar	Polar	39.6	45.8			

(a) Ablation study on coordinate system and detection head.

(b) Ablation on positional encoding (PE).

Methods	Multi-scale	mAP↑	NDS↑	mAOE↓
PolarFormer-CC-s	✗	38.1	44.9	40.0
PolarFormer-PC-s	✗	38.8	45.0	37.6
PolarFormer-s	✗	39.1	45.0	37.3
PolarFormer-CC	✓	38.1	45.0	37.5
PolarFormer-PC	✓	38.5	45.5	40.8
PolarFormer	✓	39.6	45.8	37.5

(c) Effectiveness of multi-scale polar representation.

\mathcal{N}_1	\mathcal{R}_1	mAP↑	NDS↑
240	64	38.8	45.0
256	64	39.6	45.8
272	64	38.8	45.0
256	56	38.7	45.4
256	72	38.5	45.4
256	80	38.7	45.6

(d) Ablation on polar resolution.

further improve; **(II)** Our PolarFormer-CC (with Cartesian feature and prediction) outperforms the Centerpoint equipped with specially designed CBGS head [41]; **(III)** When Polar BEV map is used to feed into a Cartesian decoder head, *on-par* performance is achieved with the post-processing counterpart; **(IV)** PolarFormer, our full model that predicts all 10 categories with one head and without any post-processing procedure, exceeds highly optimized Centerpoint by 1.1% in mAP and 0.2% in NDS. This suggests the significance of *Polar* in both representation learning and exploitation (*i.e.*, decoding).

Visualizations and analysis (I) As shown in Figure 5a and Figure 5b, compared to the Polar map, Cartesian usually downsamples the nearby area (red) with information loss, while upsamples the distant area (green) without actual information added. This would explain the inferiority of Cartesian. **(II)** With the quantitative evaluation in Figure 5c, it is evident that our model in Polar coordinate yields better results than the state-of-the-art methods in Cartesian consistently across three distances (Near/Medium/Far). Specifically, our method with Polar coordinate outperforms that of Cartesian by mAP 3.1% and NDS 0.3% in nearby area, mAP 1.3% and NDS 0.7% in medium area. Our method

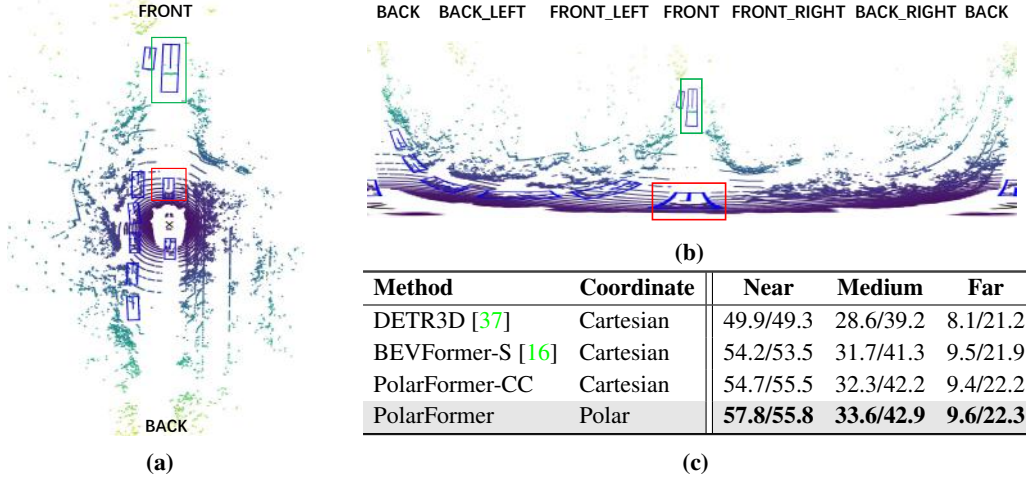


Figure 5: 3D object detection in (a) Cartesian BEV vs. (b) Polar BEV, and (c) Performance comparison (mAP/NDS) at three distances (Near/Medium/Far). Red and green boxes show the same objects in different coordinates.

Enc.	Dec.	Diff. res.	mAP↑	NDS↑	FPS↑
3	3	✗	38.8	45.4	1.6
3	3	✓	38.6	45.2	1.8
3	6	✗	38.2	44.6	1.6
3	6	✓	38.4	44.7	1.8
6	6	✗	37.9	44.8	1.4
6	6	✓	39.6	45.8	1.7

Table 4: Ablation study on model architecture.

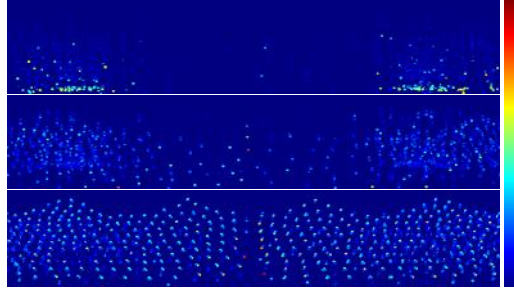


Figure 6: Multi-scale Polar BEV attention.

also outperforms DETR3D [37] and BEVFormer-S [16] at all three distances. **(III)** Figure 6 shows the attention map of the decoder query in multi-scale Polar BEV features. For better viewing, we resize the multi-scale features into the same resolution. The bottom/top corresponds to the largest/smallest scale features. The y axis represents the radius of the Polar map. It is shown that larger objects represented in the small map (top) are close to the ego car (small radius) whilst small objects in the large map (bottom) distribute through the distant area. This is highly consistent with the geometry structure of raw images (Figure 1), which has shown to be a more effective coordinate for 3D object detection as above.

Architecture (I) We first evaluate three designs of positional embedding (PE): 2D learnable PE, fixed Sine PE, and 3D PE (generated based on a set of 3D points for each Polar ray position). **(II)** As our cross-plane encoder transforms different levels of feature from FPN into Polar rays independently, we can fuse the multi-level features into a single BEV or naturally shape multiple BEVs with different or same resolutions; Table 3c clearly shows that a model with multi-scale Polar BEVs outperforms the single-scale counterpart under either coordinate. In contrast, little performance gain is achieved from multi-scale features in Cartesian. This suggests that object scale variation is a *unique* challenge with Polar, but absent with Cartesian. Our design consideration is thus verified. **(III)** Table 4 shows the results of our multi-scale variant with different numbers of Polar BEV encoder and decoder layers; We also study the effect of sharing the resolution by the multi-level features; Both contribute to the efficiency of our model; **(IV)** We study the resolution of polar map by adjusting the *azimuth* \mathcal{N}_1 and *radius* \mathcal{R}_1 (the number of Polar query in the cross-plane encoder); Table 3d shows that the *angle* with 256 and *radius* with 64 gives the best performance.

The coordinate choice for object prediction and loss optimization Until now, we have shown that making the predictions in the Polar coordinate is crucial and superior over in the Cartesian

Table 5: Ablation study on the coordinate of *object location* prediction and loss calculation.

Prediction	Loss	mAP \uparrow	NDS \uparrow
Cartesian	Cartesian	38.0	44.9
Polar	Polar	31.5	38.2
Polar	Cartesian	39.6	45.8

Table 6: Comparing the BEV semantic segmentation results in IoU on the nuScenes val set. “-” represents the unprovided result. Unless specified otherwise, we use EfficientNet-B0 [29] as the image backbone to align with OFT [27] and LSS [23]. # denotes joint learning of detection and segmentation. * denotes only training the segmentation head with the pretrained detection model frozen

Method	Multi-camera?	3D Detection		BEV semantic segmentation				
		mAP \uparrow	NDS \uparrow	Drivable	Crossing	Walkway	Carpark	Divider
VPN [21]	✗	-	-	58.0	27.3	29.4	12.3	-
PON [26]	✗	-	-	60.4	28.0	31.0	18.4	-
OFT [27]	✗	-	-	62.4	30.9	34.5	23.5	-
LSF [9]	✗	-	-	61.1	33.5	37.8	25.4	-
Image2Map [28]	✗	-	-	74.5	36.6	35.9	31.3	-
OFT [27, 9]	✓	-	-	71.7	-	-	-	18.0
LSS [23]	✓	-	-	72.9	-	-	-	20.0
Ego3RT [19]	✓	-	-	79.6	48.3	52.0	50.3	47.5
BEVFormer-S [16]	✓	-	-	80.7	-	-	-	21.3
PolarFormer	✓	-	-	81.0	48.9	55.8	52.6	42.2
BEVFormer [16]	✓	-	-	80.1	-	-	-	25.7
PolarFormer-T	✓	-	-	82.6	54.3	59.4	56.7	46.2
Ego3RT* [19]	✓	37.5	45.0	74.6	33.0	42.6	44.1	36.6
BEVFormer-S# [16]	✓	38.0	45.3	77.6	-	-	-	19.8
PolarFormer#	✓	38.8	46.5	82.6	50.1	57.4	54.1	44.5

counterpart. We conjugate that this is because with object locations projected under the Polar coordinate, a better distribution of reference points can be learned due to taking a more consistent perspective *w.r.t* the multi-camera image observation. Interestingly, we note a discrepancy in the coordinate choice between *object location* prediction and loss optimization. In particular, we find that when optimizing the object localization loss in the Polar coordinate, the loss converges very slowly and a significant performance degradation is also observed (Table 5). A plausible obstacle is the numerical discontinuity between 0 and 2π in the azimuth, which however is continuous in a physical world.

4.3 BEV segmentation

For BEV semantic segmentation, we use intersection-over-union (IoU) to assess the performance. Following LSS [23], we create a binary image prediction for each element based on a specific threshold to evaluate with the ground truth images.

We use the same backbone as [23], *i.e.*, EfficientNet-B0 [29] pre-trained on ImageNet to align with OFT [27] and LSS [23]. We use the output of the Polar BEV encoder to predict the segmentation mask in Polar coordinate. We use the initial learning rate of 3×10^{-4} and weight decay of 0.075. We train the segmentation model for 24 epochs with the AdamW optimizer and the cosine annealing learning rate scheduler. The total batch size is set to 48 across six cameras. Different to LSS [23] and BEVFormer [16], we do not evaluate the two categories of Car and Vehicle as they are already predicted by the detector. Instead, we focus on the segment categories of Crossing, Walkway and Carpark (see Figure 7).

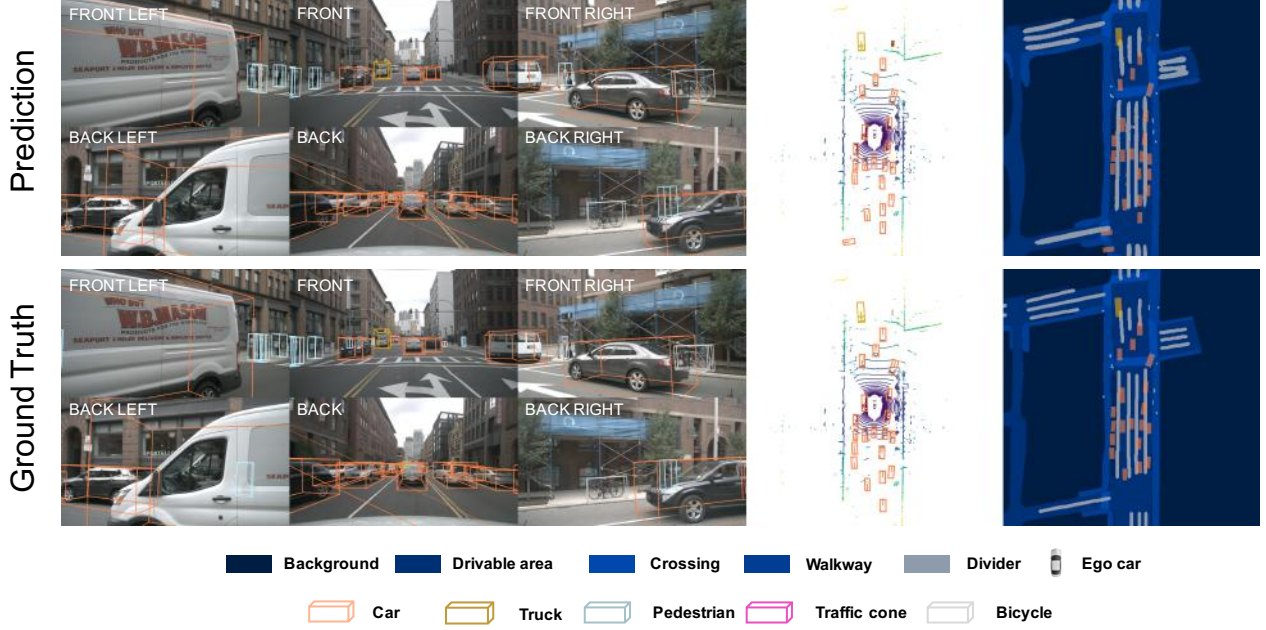


Figure 7: Visualizations: Given multi-camera images, our method is dedicated to perform 3D object detection and BEV semantic segmentation.

Table 6 summarizes the BEV semantic segmentation results on the nuScenes val set. It is evident that our method achieves the best performance among all the competitors on all the classes. In particular, this is achieved even by using a more efficient and lightweight backbone EfficientNet-B0 [29] than ResNet-101 used by BEVFormer [16]. Additionally, we equip the model with both detection and segmentation heads for multi-task joint training. In this setting, we use ResNet-101 as the backbone for more capacity. As shown in the last two rows of Table 6, our PolarFormer again outperforms ResNet-101 based BEVFormer-S in this multi-task learning setting, further indicating the superiority of conducting BEV representation based perception tasks in the Polar coordinate. Figure 7 shows the results of 3D detection and BEV segmentation.

5 Conclusions

We have proposed the Polar Transformer (PolarFormer) for 3D object detection in multi-camera 2D images from the ego car’s perspective. With a rasterized BEV Polar representation geometrically aligned to visual observation, PolarFormer overcomes irregular Polar grids by a cross-attention based decoder. Further, a multi-scale representation learning strategy is designed for tackling the intrinsic object scale variation challenge. Extensive experiments on the nuScenes dataset validate the superiority of our PolarFormer over previous alternatives on 3D object detection, as well as more generic ability on BEV semantic segmentation task.

References

- [1] Alex Bewley, Pei Sun, Thomas Mensink, Dragomir Anguelov, and Cristian Sminchisescu. Range conditioned dilated convolutions for scale invariant 3d object detection. *arXiv preprint*, 2020. 2
- [2] 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. 6, 7, 8
- [3] Yigit Baran Can, Alexander Liniger, Danda Pani Paudel, and Luc Van Gool. Structured bird’s-eye-view traffic scene understanding from onboard images. In *ICCV*, 2021. 3

- [4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 1
- [5] Qi Chen, Sourabh Vora, and Oscar Beijbom. Polarstream: Streaming object detection and segmentation with polar pillars. In *NeurIPS*, 2021. 3
- [6] Kashyap Chitta, Aditya Prakash, and Andreas Geiger. Neat: Neural attention fields for end-to-end autonomous driving. In *ICCV*, 2021. 3
- [7] MMDetection3D Contributors. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. <https://github.com/open-mmlab/mmdetection3d>, 2020. 6
- [8] Mingyu Ding, Yuqi Huo, Hongwei Yi, Zhe Wang, Jianping Shi, Zhiwu Lu, and Ping Luo. Learning depth-guided convolutions for monocular 3d object detection. In *CVPR*, 2020. 2
- [9] Isht Dwivedi, Srikanth Malla, Yi-Ting Chen, and Behzad Dariush. Bird’s eye view segmentation using lifted 2d semantic features. In *BMVC*, 2021. 10
- [10] Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3d packing for self-supervised monocular depth estimation. In *CVPR*, 2020. 7, 8
- [11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6
- [12] Anthony Hu, Zak Murez, Nikhil Mohan, Sofía Dudas, Jeffrey Hawke, Vijay Badrinarayanan, Roberto Cipolla, and Alex Kendall. Fiery: Future instance prediction in bird’s-eye view from surround monocular cameras. In *ICCV*, 2021. 3
- [13] Junjie Huang and Guan Huang. Bevdet4d: Exploit temporal cues in multi-camera 3d object detection. *arXiv preprint*, 2022. 6
- [14] Youngwan Lee, Joong-won Hwang, Sangrok Lee, Yuseok Bae, and Jongyoul Park. An energy and gpu-computation efficient backbone network for real-time object detection. In *CVPR*, 2019. 7
- [15] Qi Li, Yue Wang, Yilun Wang, and Hang Zhao. Hdmapnet: A local semantic map learning and evaluation framework. *arXiv preprint*, 2021. 3
- [16] 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. *arXiv preprint*, 2022. 1, 2, 7, 8, 9, 10, 11
- [17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 6
- [18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 7, 8
- [19] Jiachen Lu, Zheyuan Zhou, Xiatian Zhu, Hang Xu, and Li Zhang. Learning ego 3d representation as ray tracing. In *ECCV*, 2022. 1, 3, 7, 8, 10
- [20] Xinzhu Ma, Zhihui Wang, Haojie Li, Pengbo Zhang, Wanli Ouyang, and Xin Fan. Accurate monocular 3d object detection via color-embedded 3d reconstruction for autonomous driving. In *ICCV*, 2019. 2
- [21] Bowen Pan, Jiankai Sun, Ho Yin Tiga Leung, Alex Andonian, and Bolei Zhou. Cross-view semantic segmentation for sensing surroundings. *IEEE Robotics and Automation Letters*, 2020. 10
- [22] Dennis Park, Rares Ambrus, Vitor Guizilini, Jie Li, and Adrien Gaidon. Is pseudo-lidar needed for monocular 3d object detection? In *ICCV*, 2021. 7, 8
- [23] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *ECCV*, 2020. 1, 3, 10
- [24] Meytal Rapoport-Lavie and Dan Raviv. It’s all around you: Range-guided cylindrical network for 3d object detection. In *ICCV*, 2021. 2
- [25] Cody Reading, Ali Harakeh, Julia Chae, and Steven L Waslander. Categorical depth distribution network for monocular 3d object detection. In *CVPR*, 2021. 2, 3
- [26] Thomas Roddick and Roberto Cipolla. Predicting semantic map representations from images using pyramid occupancy networks. In *CVPR*, 2020. 3, 10

- [27] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. In *BMVC*, 2019. 3, 10
- [28] Avishkar Saha, Oscar Mendez Maldonado, Chris Russell, and Richard Bowden. Translating images into maps. In *ICRA*, 2022. 3, 4, 10
- [29] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. 10, 11
- [30] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, 2019. 2
- [31] 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. 4
- [32] Li Wang, Liang Du, Xiaoqing Ye, Yanwei Fu, Guodong Guo, Xiangyang Xue, Jianfeng Feng, and Li Zhang. Depth-conditioned dynamic message propagation for monocular 3d object detection. In *CVPR*, 2021. 2
- [33] Li Wang, Li Zhang, Yi Zhu, Zhi Zhang, Tong He, Mu Li, and Xiangyang Xue. Progressive coordinate transforms for monocular 3d object detection. In *NeurIPS*, 2021. 2
- [34] Tai Wang, ZHU Xinge, Jiangmiao Pang, and Dahua Lin. Probabilistic and geometric depth: Detecting objects in perspective. In *CoRL*, 2022. 1, 2, 7, 8
- [35] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Fcos3d: Fully convolutional one-stage monocular 3d object detection. In *ICCV*, 2021. 1, 2, 6, 7, 8
- [36] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *CVPR*, 2019. 2
- [37] 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. 1, 2, 6, 7, 8, 9
- [38] Enze Xie, Zhiding Yu, Daquan Zhou, Jonah Philion, Anima Anandkumar, Sanja Fidler, Ping Luo, and Jose M Alvarez. M² 2bev: Multi-camera joint 3d detection and segmentation with unified birds-eye view representation. *arXiv preprint*, 2022. 7, 8
- [39] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3d object detection from monocular images. In *CVPR*, 2018. 2
- [40] Weixiang Yang, Qi Li, Wenxi Liu, Yuanlong Yu, Yuexin Ma, Shengfeng He, and Jia Pan. Projecting your view attentively: Monocular road scene layout estimation via cross-view transformation. In *CVPR*, 2021. 3
- [41] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3d object detection and tracking. In *CVPR*, 2021. 7, 8
- [42] Yurong You, Yan Wang, Wei-Lun Chao, Divyansh Garg, Geoff Pleiss, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving. *arXiv preprint*, 2019. 2
- [43] Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, Boqing Gong, and Hassan Foroosh. Polarnet: An improved grid representation for online lidar point clouds semantic segmentation. In *CVPR*, 2020. 2
- [44] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint*, 2019. 1, 2
- [45] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In *CVPR*, 2021. 2, 3
- [46] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 5, 6

A Appendix

A.1 Visualizations

We visualize our 3D object detection and BEV semantic segmentation results in Figure 8 and Figure 9.

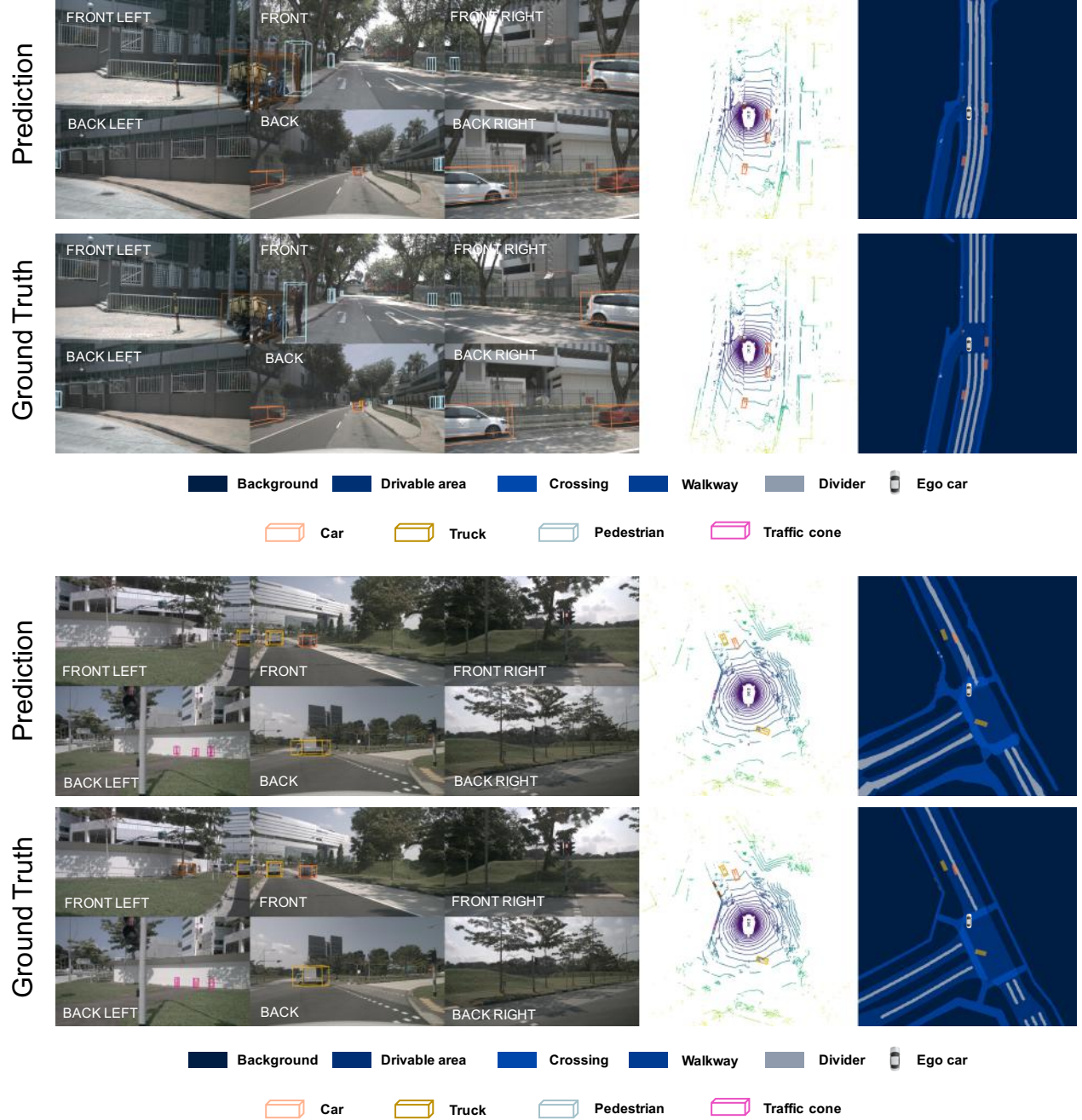


Figure 8: Visualizations (Part I): Given multi-camera images, our method dedicates to perform 3D object detection and BEV semantic segmentation.

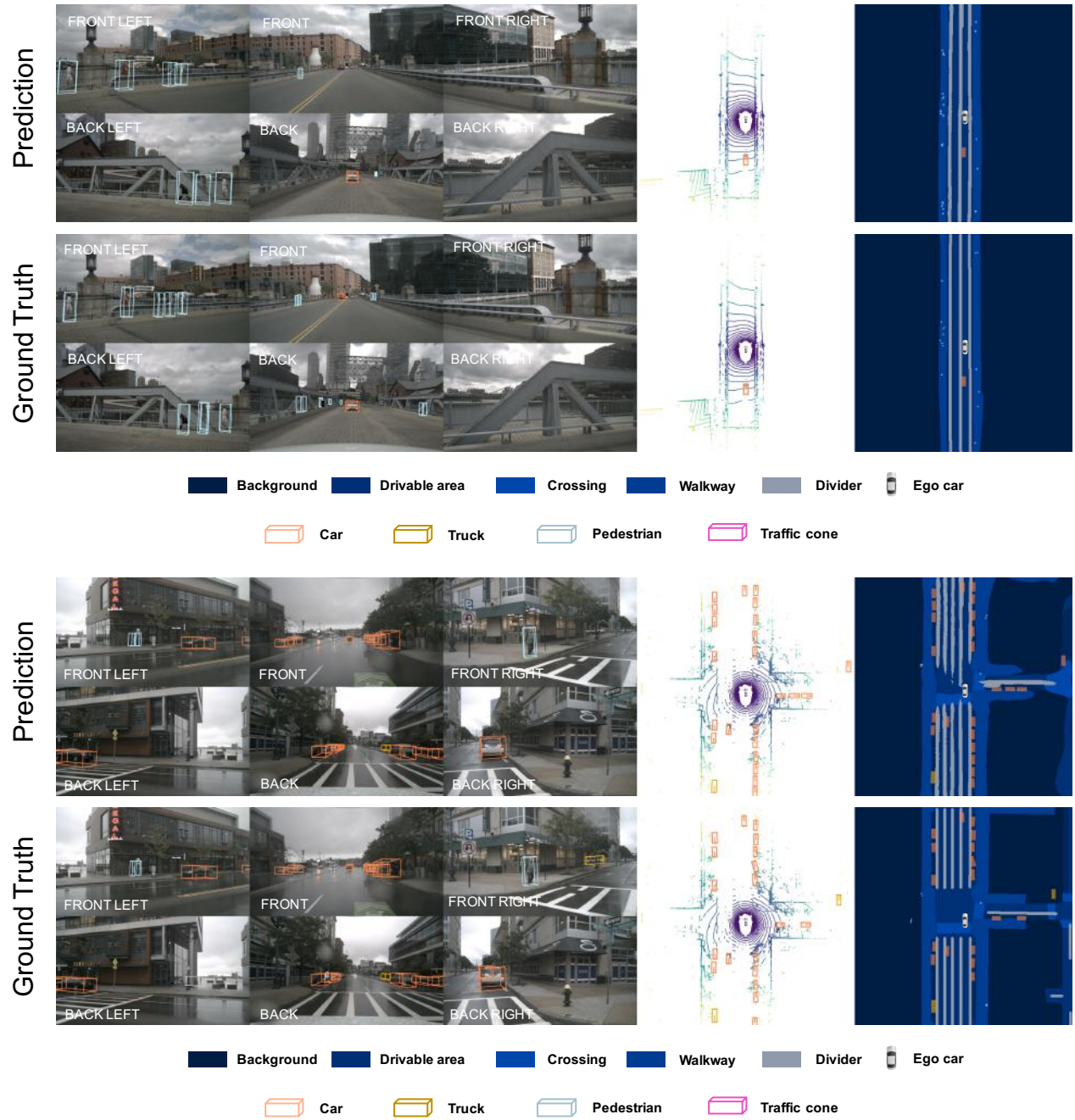


Figure 9: Visualizations (Part II): Given multi-camera images, our method dedicates to perform 3D object detection and BEV semantic segmentation.