PETRv2: A Unified Framework for 3D Perception from Multi-Camera Images

Yingfei Liu Junjie Yan Fan Jia Shuailin Li Aqi Gao
Tiancai Wang* Xiangyu Zhang Jian Sun
MEGVII Technology

Abstract

In this paper, we propose PETRv2, a unified framework for 3D perception from multi-view images. Based on PETR [24], PETRv2 explores the effectiveness of temporal modeling, which utilizes the temporal information of previous frames to boost 3D object detection. More specifically, we extend the 3D position embedding (3D PE) in PETR for temporal modeling. The 3D PE achieves the temporal alignment on object position of different frames. A feature-guided position encoder is further introduced to improve the data adaptability of 3D PE. To support for multi-task learning (e.g., BEV segmentation and 3D lane detection), PETRv2 provides a simple yet effective solution by introducing task-specific queries, which are initialized under different spaces. PETRv2 achieves state-of-the-art performance on 3D object detection, BEV segmentation and 3D lane detection. Detailed robustness analysis is also conducted on PETR framework. We hope PETRv2 can serve as a strong baseline for 3D perception. Code is available at https://github.com/megvii-research/PETR.

1. Introduction

Recently, 3D perception from multi-camera images for autonomous driving system has drawn a great attention. The multi-camera 3D object detection methods can be divided into BEV-based [11, 12] and DETR-based [20, 24, 39] approaches. BEV-based methods (e.g., BEVDet [12]) explicitly transform the multi-view features into bird-eye-view (BEV) representation by LSS [33]. Different from these BEV-based countparts, DETR-based approaches [39] models each 3D object as an object query and achieve the end-to-end modeling with Hungarian algorithm [16]. Among these methods, PETR [24], based on DETR [4], converts the multi-view 2D features to 3D position-aware features by adding the 3D position embedding (3D PE). The object query, initialized from 3D space, can directly perceive the 3D object information by interacting with the produced 3D position-aware features. In this paper, we aim to build a strong and unified framework by extending the PETR with temporal modeling and the support for multi-task learning.

For temporal modeling, the main problem is how to align the object position of different frames in 3D space. Existing works [11, 20] solved this problem from the perspective of feature alignment. For example, BEVDet4D [11] explicitly aligns the BEV feature of previous frame with current frame by pose transformation. However, PETR implicitly encodes the 3D position into the 2D image features and fails to perform the explicit feature transformation. Since PETR has demonstrated the effectiveness of 3D PE (encoding the 3D coordinates into 2D features) in 3D perception, we wonder if 3D PE still works on temporal alignment. In PETR, the meshgrid points of camera frustum space, shared for different views, are transformed to the 3D coordinates by camera parameters. The 3D coordinates are then input to a simple multi-layer perception (MLP) to generate the 3D PE. In our practice, we find that PETR works well under temporal condition by simply aligning the 3D coordinates of previous frame with the current frame.

For multi-task learning, BEVFormer [20] provides a unified solution. It defines each point on BEV map as one BEV query. Thus, the BEV query can be employed for 3D object detection and BEV segmentation. However, the number of BEV query (e.g., >60,000) tends to be huge when the resolution of BEV map is relatively larger (e.g., 256 × 256). Such definition on object query is obviously not suitable for PETR due to the global attention employed in transformer decoder. In this paper, we design a unified sparse-query solution for multi-task learning. For different tasks, we define sparse task-specific queries under different spaces. For example, the lane queries for 3D lane detection are defined in 3D space with the style of anchor lane while seg queries for BEV segmentation are initialized under the BEV space. Those sparse task-specific queries are input to the same transformer decoder to update their representation and further injected into different task-specific heads to produce high-quality predictions.

*Corresponding author

Besides, we also improve the generation of 3D PE and provide a detailed robustness analysis on PETRv2. As mentioned above, 3D PE in PETR is generated based on the fixed meshgrid points in camera frustum space. All images from one camera view share the 3D PE, making 3D PE data-independent. In this paper, we further improve the original 3D PE by introducing a feature-guided position encoder (FPE). Concretely, the projected 2D features are firstly injected into a small MLP network and a Sigmoid layer to generate the attention weight, which is used to reweight the 3D PE in an element-wise manner. The improved 3D PE is data-dependent, providing the informative guidance for the query learning in transformer decoder. For comprehensive robustness analysis on PETRv2, we consider multiple noise cases, including the camera extrinsics noise, camera miss and time delay.

To summarize, our contributions are:

- We study a conceptually simple extension of position embedding transformation to temporal representation learning. The temporal alignment can be achieved by the pose transformation on 3D PE. A feature-guided position encoder is further proposed to reweight the 3D PE with the guidance from 2D image features.
- A simple yet effective solution is introduced for PETR to support the multi-task learning. BEV segmentation and 3D lane detection are supported by introducing task-specific queries.
- Experiments show that the proposed framework achieves state-of-the-art performance on both 3D object detection, BEV segmentation and 3D lane detection. Detailed robustness analysis is also provided for comprehensive evaluation on PETR framework.

2. Related Work

2.1. Multi-View 3D Object Detection

Previous works [2,6,13–15,29,35,37,38] perform 3D object detection mainly under the mono setting. Recently, 3D object detection based on multi-view images has attracted more attention. ImVoxelNet [34] and BEVDet [12] projected the multi-view image features into BEV representation. Then the 3D object detection can be performed using the methods from 3D point cloud, like [42]. DETR3D [39] and PETR [24] conduct the 3D object detection mainly inspired by the end-to-end DETR methods [4,23,28,46]. The object queries are defined in 3D space and interact with the multi-view image features in transformer decoder. BEVFormer [20] further introduces the temporal information into vision-based 3D object detection. The spatial cross-attention is adopted to aggregate image features, while the temporal self-attention is used to fuse the history BEV features. BEVDet4D [11] extends the BEVDet [12] by the temporal modeling and achieves good speed estimation. Both BEVFormer [20] and BEVDet4D [11] align the multi-frame features in BEV space. Different from them, we extend the temporal version from PETR and achieve the temporal alignment from the perspective of 3D position embedding (3D PE).

2.2. BEV Segmentation

BEV segmentation focus on the perception in the BEV view. It takes the multi-view images as input and rasterizes output onto a map view. VPN [30] proposes a view parsing network under the simulated environments and then transfers it to real-world environments to perform cross-view semantic segmentation. LSS [33] transforms the 2D features into 3D space by implicit estimation of depth and employs different heads for BEV segmentation and planning. M^{2}BEV [40] further uses the camera parameters to project the features extracted from backbone to the 3D ego-car coordinate to generate the BEV representation. Then multi-task heads are used for 3D detection and segmentation. BEVFormer [20] generates the BEV features from multi-camera inputs by interacting the predefined grid-shaped BEV queries with the 2D image features. CVT [43] uses cross-view transformer to learn geometric transformation implicitly. HDMapNet [19] transforms multi-view images to the BEV view and produces a vectorized local semantic map. BEVSegFormer [32] proposes multi-camera deformable attention to construct semantic map.

2.3. 3D Lane Detection

BEV segmentation can reconstruct the elements of local map. However, it fails to model the spatial association between different instances. Recently, the 3D lane detection task has attracted more and more attention. 3D-LaneNet [7] is the first method that makes the 3D lane prediction. It uses inverse perspective mapping (IPM) to transform feature from front view to BEV. Gen-LaneNet [8] introduces a new anchor lane representation to align the perspective anchor representation and BEV feature. Persformer [5] employs the deformable attention to generate BEV features by attending local context around reference points. CurveFormer [1] introduces a curve cross-attention module to compute the similarities between curve queries and image features. It employs deformable attention to obtain the image features corresponding to the reference points.

3. Method

3.1. Overall Architecture

As illustrated in Fig. 1, the overall architecture of PETRv2 is built upon the PETR [24] and extended with temporal modeling and BEV segmentation. The 2D image features are extracted from multi-view images with the

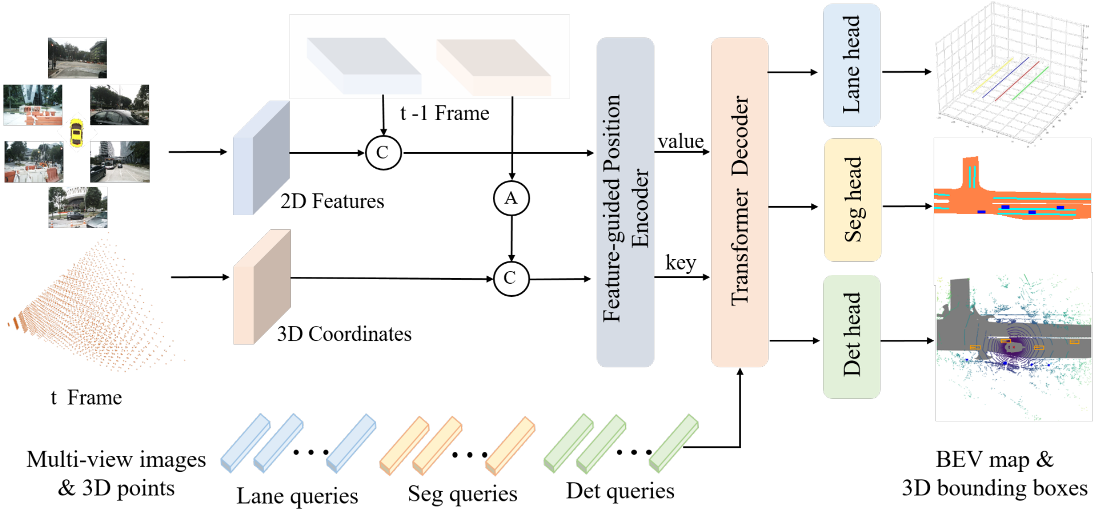


Figure 1. The paradigm of the proposed PETRv2. The 2D features are extracted by the backbone network from the multi-view images and the 3D coordinates are generated following the same way as PETR [24]. To achieve the temporal alignment, the 3D coordinates in PETR of previous frame $t-1$ are firstly transformed through pose transformation. Then 2D image features and 3D coordinates of two frames are concatenated together and injected to feature-guided position encoder to generate the key and value components for the transformer decoder. The detection, segmentation and lane queries, initialized under different spaces, interact with the key and value components in transformer decoder. The updated queries are further used to predict the 3D bounding boxes, BEV segmentation map and the 3D lanes with task-specific heads. Ⓐ is 3D coordinates alignment from frame $t-1$ to frame t. Ⓒ is concatenation operation along the batch axis.

2D backbone (e.g., ResNet-50), and the 3D coordinates are generated from camera frustum space as described in PETR [24]. Considering the ego motion, 3D coordinates of the previous frame $t-1$ are first transformed into the coordinate system of current frame t through the pose transformation. Then, the 2D features and 3D coordinates of adjacent frames are respectively concatenated together and input to the feature-guided position encoder (FPE). After that, the FPE is employed to generate the key and value components for the transformer decoder. Further, task-specific queries including the detection queries (det queries) and segmentation queries (seg queries), which are initialized from different spaces, are fed into the transformer decoder and interact with multi-view image features. Lastly, the updated queries are input to the task-specific heads for final prediction.

3.2. Temporal Modeling

PETR [24] leverages image features and projected 3D points to generate implicit 3D features for multi-view 3D detection. In this section, we extend it with the temporal modeling, which is realized by a 3D coordinates alignment (CA) for better localization and speed estimation.

3D Coordinates Alignment The temporal alignment is to transform the 3D coordinates of frame $t-1$ to the coordinate system of frame t (see Fig. 2). For clarity, we first denote some coordinate systems: camera coordinate as $c(t)$, lidar coordinate as $l(t)$, and ego coordinate as $e(t)$ at frame t. What's more, global coordinates as g. We define T^{dst}_{src} as the transformation matrix from the source coordinate system to the target coordinate system.

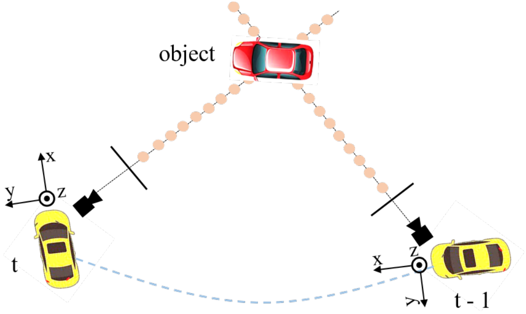


Figure 2. The illustration of the coordinate system transformation from frame $t-1$ to frame t.

We use $l(t)$ as the default 3D space for multi-view camera 3D position-aware feature generation. The 3D points $P^{l(t)}_i(t)$ projected from i-th camera can be formulated as:

$$P^{l(t)}_i(t) = T^{l(t)}_{c_i(t)} K^{-1}_i P^m(t) \tag{1}$$

where $P^m(t)$ is the points set in the meshgrid of camera frustum space at frame t. $K_i \in R^{4\times4}$ is the camera intrinsic matrix of the i-th camera. Given the auxiliary frame $t-1$, we align the coordinates of 3D points from frame $t-1$ to

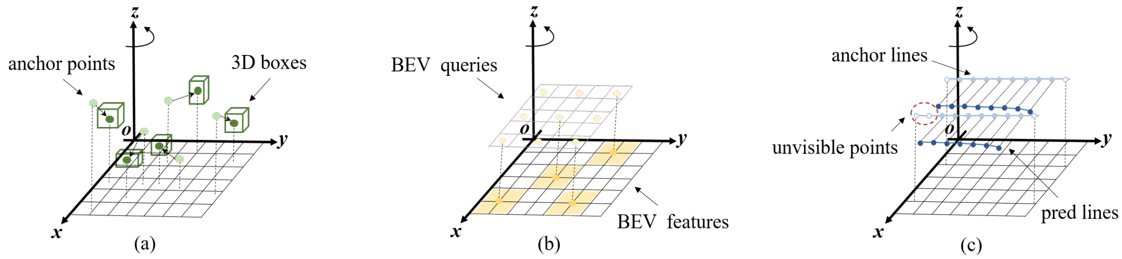


Figure 3. The definition of three kinds of queries for multi-task learning. The det query is defined in the whole 3D space while the seg query is initialized under the BEV space. The lane query is defined with the anchor line, which is constructed with 300 anchor points.

frame t:

$$P_i^{l(t)}(t-1) = T_{l(t-1)}^{l(t)} P_i^{l(t-1)}(t-1) \tag{2}$$

With $global$ coordinate space acting as a bridge between frame $t-1$ and frame t, $T_{l(t-1)}^{l(t)}$ can be easily calculated:

$$T_{l(t-1)}^{l(t)} = T_{e(t)}^{l(t)} T_g^{e(t)} {T_g^{e(t-1)}}^{-1} {T_{e(t-1)}^{l(t-1)}}^{-1} \tag{3}$$

The aligned point sets $[P_i^{l(t)}(t-1), P_i^{l(t)}(t)]$ are used to generate the 3D PE, as described in Sec. 3.4.

3.3. Multi-task Learning

In this section, we aim to equip PETR [24] with seg queries and lane queries to support high-quality BEV segmentation and 3D Lane detection.

BEV Segmentation A high-resolution BEV map can be partitioned into a small number of patches. We introduce the seg query for BEV segmentation and each seg query corresponds to a specific patch (e.g., top-left 25×25 pixels of the BEV map). As shown in Fig. 3 (b), the seg queries are initialized with fixed anchor points in BEV space, similar to the generation of detection query (det query) in PETR. These anchor points are then projected into the seg queries by a simple MLP with two linear layers. After that, the seg queries are input to the transformer decoder and interact with the image features. For the transformer decoder, we use the same framework as detection task. Then the updated seg queries are finally fed into the segmentation head, similar to the decoder in CVT [43], to predict the final segmentation results. We use focal loss to supervise the predictions of each category separately.

3D Lane Detection We add lane queries on PETR to support 3D lane detection (see Fig. 3 (c)). We define the 3D anchor lanes, each of which is represented as an ordered set of 3D coordinates: $l = \{(x_1, y_1, z_1,), (x_2, y_2, z_2), \cdots, (x_n, y_n, z_n)\}$, where n is the number of the sample points of each lane. In order to improve the prediction ability for 3D lanes, we use a fixed sampling point set uniformly sampled along the Y-axis, similar to Persformer [5]. Different from Persformer, our anchor lanes are parallel to the Y-axis while the Persformer predefines different slopes for each anchor line. The updated lane queries from transformer decoder are used to predict the 3D lane instances. The 3D lane head predicts the lane class C as well as the relative offset $(\Delta x, \Delta z)$ along x-axis and z-axis compared to the anchor lanes. Since the length of 3D lane is not fixed, we also predict the visibility vector T_{vis} of size n to control the start and end points of the lane. We use focal loss to supervise the predictions of the lane category and visibility. We also use $L1$ loss to supervise the predictions of the offset.

3.4. Feature-guided Position Encoder

PETR transforms the 3D coordinates into 3D position embedding (3D PE). The generation of 3D position embedding can be formulated as:

$$PE_i^{3d}(t) = \psi(P_i^{l(t)}(t)) \tag{4}$$

where $\psi(.)$ is a simple multi-layer perception (MLP). The 3D PE in PETR is independent with the input image. We argue that the 3D PE should be driven by the 2D features since the image feature can provide some informative guidance (e.g., depth). In this paper, we propose a feature-guided position encoder, which implicitly introduces vision prior. The generation of feature-guided 3D position embedding can be formulated as:

$$PE_i^{3d}(t) = \xi(F_i(t)) * \psi(P_i^{l(t)}(t)) \tag{5}$$

where ξ is also a small MLP network. $F_i(t)$ is the 2D image features of the i-th camera. As illustrated in Fig. 4, the 2D image features projected by a 1×1 convolution are fed into a small MLP network ξ and Sigmoid function to obtain the attention weights. The 3D coordinates are transformed by another MLP network ψ and multiplied with the attention weights to generate the 3D PE. The 3D PE is added with 2D features to obtain the key value for transformer decoder. The projected 2D features are used as the value component for transformer decoder.

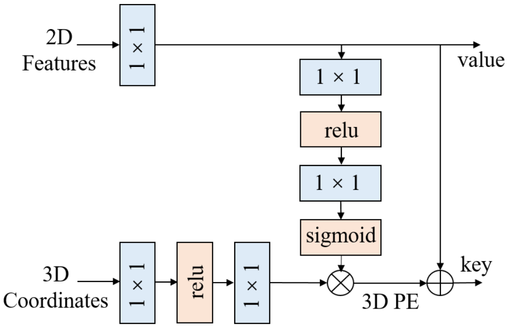


Figure 4. Architecture of feature-guided position encoder. Different from PETR [24], 3D PE in PETRv2 is generated in a data-dependent way and guided by the image features.

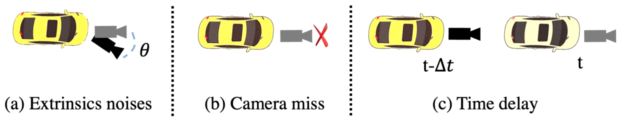
3.5. Robustness Analysis

Though recently there are lots of works on autonomous driving systems, only a few works [20, 33] explore the robustness of proposed methods. LSS [33] presents the performance under extrinsics noises and camera dropout at test time. Similarly, BEVFormer [20] demonstrates the robustness of model variants to camera extrinsics. In practice, there are diverse sensor errors and system biases, and it is important to validate the effect of these circumstances due to the high requirements of safety and reliability. We aim to give an extensive study of our method under different conditions. As shown in Fig. 5, we focus on three common types of sensor errors as follows:

Extrinsics noise: Extrinsics noises are very common in reality, such as the camera shake caused by a car bump or camera offset by the environmental forces. In these cases, extrisics provided by the system is not that accurate and the perception results will be affected.

Camera miss: Camera image miss occurs when one camera breaks down or is occluded. Multiview images provide panoramic visual information, yet the possibility exists that one of them is absent in the real world. It is necessary to evaluate the importance of these images so as to prepare the strategy of sensor redundancy in advance.

Camera time delay: Camera time delay is also a challenge due to the camera exposure time, especially in night. The long exposure time causes the system is fed with images from the previous time, and brings the significant output offsets.



(a) Extrinsics noises (b) Camera miss (c) Time delay

Figure 5. We analyze the system robustness of PETR series under three simulated sensor errors: (a) extrinsics noise, (b) camera miss and (c) camera time delay.

4. Experiments

4.1. Datasets and Metrics

We evaluate our approach on nuScenes benchmark [3] and OpenLane benchmark [5]. NuScenes [3] is a large-scale multi-task dataset covering 3D object detection, BEV segmentation, 3D object tracking, etc. The dataset is officially divided into training/validation/testing sets with 700/150/150 scenes, respectively. We mainly focus on two sub-tasks: 3D object detection and BEV segmentation. We also conduct the 3D lane detection experiments on Open-Lane benchmark [5]. Openlane [5] is a large-scale real world 3D lane dataset. It has 200K frames and over 880K carefully annotated lanes and covers a wide range of lane types using 14 lane categories.

For 3D object detection, each scene has 20s video frames and is annotated around 40 key frames. We report the official evaluation metrics including nuScenes Detection Score (NDS), mean Average Precision (mAP), and five True Positive (TP) metrics: mean Average Translation Error (mATE), mean Average Scale Error (mASE), mean Average Orientation Error(mAOE), mean Average Velocity Error(mAVE), mean Average Attribute Error(mAAE). NDS is a comprehensive indicator to evaluate the detection performance.

For BEV segmentation, we follow LSS [33] and use IoU score as the metric. The ground-truth includes three different categories: Driveable area, Lane and Vehicle. The lane category is formed by two map layers: lane-Divider and Road-Divider. For Vehicle segmentation, we obtain the BEV ground truth by projecting 3D bounding boxes into the BEV plane [33]. The Vehicle segmentation ground truth refers to all bounding boxes of meta-category Vehicle, which contains bicycle, bus, car, construction, motorcycle, trailer and truck.

For 3D lane detection, we follow Persformer [5] using F1-Score and category accuracy as the metrics. When 75% points of a predicted lane instance have the point-wise euclidean distance less than 1.5 meters, the lane instance is considered to be correctly predicted. We also report X error near, X error far, Z error near, Z error far to evaluate the models. These four metrics are used to evaluate the average error of the results in specified ranges.

4.2. Implementation Details

In our implementation, ResNet [9], VoVNetV2 [17] and EfficientNet [36] are employed as the backbone for feature extraction. The P4 feature (merging the C4 and C5 features from backbone) with 1/16 input resolution is used as the 2D feature. The generation of 3D coordinates is consistent with PETR [24]. Following BEVDet4D [11], we randomly sample a frame as previous frame ranging from $[3T, 27T]$ during training, and sample the frame at $15T$ during inference. $T(\approx 0.083)$ is the time interval between two sweep

Table 1. Comparison of recent works on the nuScenes val set. The results of FCOS3D and PGD are fine-tuned and tested with test time augmentation. The DETR3D, BEVDet and PETR are trained with CBGS [45]. † is initialized from a FCOS3D backbone.

Methods	Backbone	Size	NDS↑	mAP↑	mATE↓	mASE↓	mAOE↓	mAVE↓	mAAE↓
CenterNet [44]	DLA	-	0.328	0.306	0.716	0.264	0.609	1.426	0.658
FCOS3D [38]	Res-101	1600×900	0.415	0.343	0.725	0.263	0.422	1.292	**0.153**
PGD [37]	Res-101	1600×900	0.428	0.369	0.683	0.260	0.439	1.268	0.185
BEVDet [12]	Swin-T	1408×512	0.417	0.349	0.637	0.269	0.490	0.914	0.268
DETR3D† [39]	Res-101	1600×900	0.434	0.349	0.716	0.268	0.379	0.842	0.200
PETR† [24]	Res-101	1600×900	0.442	0.370	0.711	0.267	0.383	0.865	0.201
BEVFormer† [20]	Res-101	1600×900	0.517	0.416	0.673	0.274	0.372	0.394	0.198
BEVDet4D [11]	Swin-B	1600×640	0.515	0.396	**0.619**	**0.260**	0.361	0.399	0.189
PETRv2	Res-50	800×320	0.456	0.350	0.726	0.277	0.505	0.503	0.181
PETRv2	Res-50	1600×640	0.494	0.398	0.690	0.273	0.467	0.424	0.195
PETRv2†	Res-101	800×320	0.489	0.375	0.677	0.271	0.414	0.435	0.192
PETRv2†	Res-101	1600×640	**0.524**	**0.421**	0.681	0.267	**0.357**	**0.377**	0.186

frames. Our model is trained using AdamW [27] optimizer with a weight decay of 0.01. The learning rate is initialized with 2.0×10^{-4} and decayed with cosine annealing policy [26]. All the experiments are trained for 24 epochs (2× schedule) on 8 Tesla A100 GPUs with a total batch size of 8 except for the ablation study. No test time augmentation methods are used during inference.

For 3D object detection, we perform experiments with 1500 det queries on nuScenes test dataset. Following the settings in PETR [24], we initialize a set of learnable anchor points in 3D world space, and generate these queries through a small MLP network. Similar to FCOS3D [38], we add extra disentangled layers for regression targets. We extend query denoise of DN-DETR [18] to accelerate convergence of 3D object detection. For each ground-truth 3D box, the center is shifted by a random noise less than (w/2, l/2, h/2), where (w, l, h) is the size of object. We also adopt the focal loss [21] for classification and $L1$ loss for 3D bounding box regression. The Hungarian algorithm [16] is used for label assignment between ground-truths and predictions. For BEV segmentation, we follow the settings in [33]. We use the map layers provided by the nuScenes dataset to generate the 200 × 200 BEV map ground truth. We set the patch size to 25 × 25 and 625 seg queries are used to predict the final BEV segmentation result. For 3D lane detection, we follow the settings in [5]. The input size of images is 360 × 480. We use 100 lane queries to predict the 3D lanes. We set the number of points in each anchor lane to 10 and the prediction range is $[3m, 103m]$ on Y-axis and $[-10m, 10m]$ on X-axis. The distance is calculated at several fixed positions along the Y-axis: [5, 10, 15, 20, 30, 40, 50, 60, 80, 100] for 3D anchor lanes.

To simulate extrinsic noises and evaluate the effect, we choose to randomly apply 3D rotation to camera extrinsics. 3D rotation is very common and typical in real scenarios, and we ignore other noisy patterns such as translation to avoid multi-variable interference. Specifically, we randomly choose one from multiple cameras to apply 3D rotation. Denoting α, β, γ as angles (in degree) along X, Y, Z axes respectively, we investigate in several rotation settings with maximum amplitudes $\alpha_{max}, \beta_{max}, \gamma_{max} \in \{2, 4, 6, 8\}$, where $\alpha_{max} = 2$ means that α is uniformly sampled from $[-2, 2]$, for example. In experiment, we use $R_{max} = M$ to denote $\alpha_{max} = \beta_{max} = \gamma_{max} = M$.

4.3. State-of-the-art Comparison

Tab. 1 compares the performance with recent works on nuScenes val set. Our method achieves state-of-the-art performance among public methods. PETRv2 achieves 39.8% mAP and 49.4% NDS even with ResNet-50. Tab. 2 shows the performance comparison on nuScenes test set. Our PETRv2 with VoVNet surpasses the PETR by a large margin (8.3% NDS and 6.7% mAP). Benefiting from the temporal modeling, the mAVE can achieved with 0.343m/s compared to the 0.808m/s of PETR. When compared with other temporal methods, PETRv2 surpasses the BEVDet4D [11] with Swin-Base [25] and BEVFormer [20] V2-99 [17] by 2.2% NDS. It shows that the temporal alignment by 3D PE can also achieve remarkable performance. It should be noted that PETRv2 can be easily employed for practical application without the explicit feature alignment.

We also compare the BEV segmentation performance on nuScenes dataset. As shown in Tab. 3, we conduct the experiments with ResNet-101 and VoV-99 backbones. Since PETRv2 is the temporal extension of PETR so we mainly compare the performance with BEVFormer for fair comparison. With ResNet-101 backbone, our PETRv2 outperforms BEVFormer on IoU-lane and IoU-Drive metrics by a large margin and achieves comparable performances on IoU-Vehicle metric. With the pretrained VoV-99 backbone,

Table 2. Comparison of recent works on the nuScenes test set. ∗ are trained with external data. ‡ is test time augmentation. "ms " indicates using the resolution of 800 × 320 and 1600 × 640 as the inputs.

Methods	Backbone	NDS↑	mAP↑	mATE↓	mASE↓	mAOE↓	mAVE↓	mAAE↓
CenterNet [44]	DLA	0.400	0.338	0.658	0.255	0.629	1.629	0.142
FCOS3D‡ [38]	Res-101	0.428	0.358	0.690	0.249	0.452	1.434	0.124
PGD‡ [37]	Res-101	0.448	0.386	0.626	0.245	0.451	1.509	0.127
DD3D∗‡ [31]	V2-99	0.477	0.418	0.572	0.249	0.368	1.014	0.124
DETR3D∗ [39]	V2-99	0.479	0.412	0.641	0.255	0.394	0.845	0.133
BEVDet [12]	Swin-S	0.463	0.398	0.556	0.239	0.414	1.010	0.153
BEVDet∗ [12]	V2-99	0.488	0.424	0.524	0.242	0.373	0.950	0.148
M^{2}BEV [40]	X-101	0.474	0.429	0.583	0.254	0.376	1.053	0.190
PETR∗ [24]	V2-99	0.504	0.441	0.593	0.249	0.383	0.808	0.132
BEVFormer [20]	Res-101	0.535	0.445	0.631	0.257	0.405	0.435	0.143
BEVFormer∗ [20]	V2-99	0.569	0.481	0.582	0.256	0.375	0.378	0.126
BEVDet4D‡ [11]	Swin-B	0.569	0.451	**0.511**	**0.241**	0.386	**0.301**	0.121
PETRv2	Res-101	0.553	0.456	0.601	0.249	0.391	0.382	0.123
PETRv2∗	V2-99	0.582	0.490	0.561	0.243	0.361	0.343	0.120
PETRv2∗ ms	V2-99	**0.591**	**0.508**	0.543	**0.241**	**0.360**	0.367	**0.118**

Table 3. Comparison of recent BEV segmentation works on the nuScenes val set. ∗ are trained with external data. The performance of M^{2}BEV is reported with X-101 [41] backbone.

Methods	Backbone	Drive	Lane	Vehicle
Lift-Splat [33]	Res-101	0.729	0.200	0.321
FIERY [10]	Res-101	-	-	0.382
M^{2}BEV [40]	X-101	0.759	0.380	-
BEVFormer [20]	Res-101	0.801	0.257	0.448
PETRv2	Res-101	0.833	0.448	0.434
PETRv2∗	V2-99	**0.856**	**0.490**	**0.463**

Table 4. Comparison of recent 3D lane detection works on OpenLane benchmark. PETRv2-V and PETRv2-E are our method with VoVNetV2 [17] and EfficientNet [36] backbones. ∗ is our method with 400 anchor points. The performance of Persformer is reported with EfficientNet [36] backbone. ‡ denotes projecting 2D lane results from CondLaneNet [22] to BEV using IPM.

Methods	F-score(%)	X-near	X-far	Z-near	Z-far
3D-LaneNet [7]	44.1	0.479	0.572	0.367	0.443
Gen-LaneNet [8]	32.3	0.591	0.684	0.411	0.521
Cond-IPM‡	36.6	0.563	1.080	0.421	0.892
PersFormer [5]	50.5	0.485	**0.553**	0.364	0.431
PETRv2-E	51.9	0.493	0.643	0.322	0.463
PETRv2-V	57.8	0.427	0.582	0.293	0.421
PETRv2-V∗	**61.2**	**0.400**	0.573	**0.265**	**0.413**

our PETRv2 achieves state-of-the-art performance.

As shown in Tab. 4, we compare the performance with other state-of-the-art 3D lane detection methods. Since Persformer [5] with EfficientNet backbone is a static method, we do not use the temporal information for fair comparison. With the same EfficientNet backbone, our method achieves 51.9% F1-score compared to the 50.5% in Performer. With the strong pretrained VoV-99 backbone, the performance of our method is greatly improved. We also try to represent each lane with 400 anchor points and the experimental result shows that increasing the number of anchor points leads to further performance improvements. We argue that 10 anchor points are not enough to model a relatively complex 3D lane, making it difficult to make accurate prediction. It should be noted that the large number of anchor points only increase marginal computation cost in our method. The increased cost is mainly from the higher dimension of the MLP in the lane head.

4.4. Ablation Study

In this section, we conduct the ablations with VoVNet-99 backbone. The backbone is pretrained on DDAM15M dataset [31] and train set of Nuscenes [3]. The input image size is 800 × 320 and the model is trained with 24 epochs. The number of detection queries is set to 900.

Here we explore the effect of two key components in our design: 3D coordinates alignment (CA) and feature-guided position encoder (FPE). For the ablation study, we only trained the 3D detection branch for clarity. As shown in Tab. 5(a), without CA, PETRv2 only improves the performance by 2.7% NDS and 0.5% mAP. With CA, the performance is further improved by 2.1% NDS and 0.9% mAP. The mAVE metric is decreased to 0.429 m/s, which shows a large margin compared to the original PETR baseline. To verify the effectiveness of FPE, we replace the 3D position encoder in PETR with FPE. The NDS metric is increased by 1.5% while mAP is only increased by 0.2%. When we

Table 5. The impact of 3D coordinates alignment and feature-guided position encoder. Here, CA is the 3D coordinates alignment and FPE is the proposed feature-guided position encoder.

	CA	FPE	NDS↑	mAP↑	mATE↓	mASE↓	mAOE↓	mAVE↓	mAAE↓
PETR			0.434	0.379	0.754	0.272	0.476	0.838	0.211
PETR		✓	0.449	0.381	0.749	0.271	0.462	0.736	0.200
PETRv2			0.461	0.384	0.775	0.270	0.470	0.605	0.189
PETRv2	✓		0.482	0.393	0.774	0.272	0.486	0.429	0.187
PETRv2	✓	✓	**0.496**	**0.401**	**0.745**	**0.268**	**0.448**	**0.394**	**0.184**

apply the FPE on PETRv2, the mAP achieves a relatively higher improvement (0.8%). It indicates that FPE module is also beneficial to the temporal version of PETR.

4.5. Robustness analysis

Tab. 6 reports a summary of quantitative results on the nuScenes dataset with extrinsics noises during inference. We compare PETRv2, PETR and PETR + FPE (FPE denotes the feature-guided position encoder). As the noise increases, the performance of all three models decreases continually, indicating the impact of extrinsics noises. In the extreme noise setting $R_{max} = 8$, PETRv2 drops 4.12% mAP and 2.85% NDS, PETR+FPE drops 4.68% mAP and 3.42% NDS, while PETR drops 6.33% mAP and 4.54% NDS. We observe that FPE improves the robustness to extrinsics noises, while temporal extension with multiple frames does not bring significant robustness gains.

Table 6. Quantitative results on the nuScenes val set with extrinsics noises. The metrics in each cell are mAP[%]. $R_{max} = M$ denotes the maximum angle of three axes is M in degree.

Methods	$R_{max} = 2$	$R_{max} = 4$	$R_{max} = 6$
PETR	36.71 (↓1.16)	34.58 (↓3.29)	32.79 (↓5.08)
PETR+FPE	37.17 (↓0.96)	35.83 (↓2.30)	34.47 (↓3.66)
PETRv2	39.13 (↓0.95)	37.69 (↓2.15)	36.66 (↓3.42)

We also show how the model performs when randomly losing one camera in Fig. 6. Among these six cameras of nuScenes dataset, the front and back cameras are the most important ones, and their absences leads to a drop of 5.05% and 13.19% mAP, respectively. The back camera is especially essential due to its large field of view (120°). Losing other cameras also brings an average performance decrease of 2.93% mAP and 1.93% NDS. Note that the overlap region between cameras is very small for the nuScenes dataset, thus performance drop caused by any camera miss is hard to be compensated by adjacent ones. In practice, sensor redundancy is necessary in case of emergency and complementary of cameras requires deeper explorations.

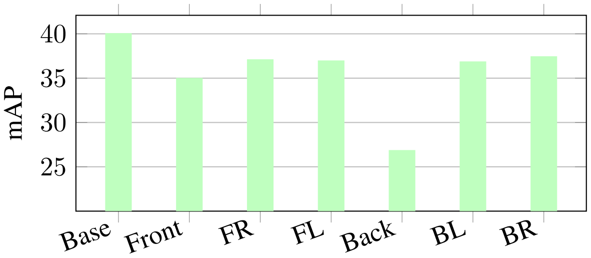


Figure 6. The performance on nuScenes val when losing each of camera images. FR, FL, BL and BR denote the front-right, front-left, back-left and back-right, respectively.

The effect of camera time delay is demonstrated in Tab. 7. In nuScenes, key frames are annotated with ground-truth, and we leverage unannotated frames between key frames as input images to simulate the time delay. The delay of 0.083s leads to a drop of 3.19% mAP and 8.4% NDS, indicating the significant impact of time delay. When time delay increase to over 0.3s, the performance sharply decreases to 26.08% mAP and 36.54% NDS. Since time delay is inevitable in real-world systems and affects detection a lot, more attention is supposed to pay to it.

Table 7. The performance impact (on mAP metric) of camera time delay. Here, the time delay unit $T \approx 0.083$s.

Time delay	T	2T	3T
PETRv2	36.89 (↓3.19)	33.99 (↓6.09)	30.91 (↓9.17)

5. Conclusion

In this paper, we introduce PETRv2, a unified framework for 3D perception from multi-camera images. PETRv2 extends the PETR baseline with temporal modeling and multi-task learning. With the temporal alignment on 3D position embedding, PETRv2 naturally achieves the multi-frame modeling and improves the 3D detection performance. For a fully understanding of PETRv2 framework, we further provide a detailed analysis on the robustness of PETRv2 under three types of simulated sensor errors. We hope PETRv2 can serve as a strong baseline and a unified framework for 3D perception. In the near future, we amy explore large-scale pretraining, more 3D vision tasks and multi-modal fusion for autonomous driving system.

References

[1] Yifeng Bai, Zhirong Chen, Zhangjie Fu, Lang Peng, Pengpeng Liang, and Erkang Cheng. Curveformer: 3d lane detection by curve propagation with curve queries and attention. *arXiv preprint arXiv:2209.07989*, 2022. 2

[2] Garrick Brazil and Xiaoming Liu. M3d-rpn: Monocular 3d region proposal network for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9287–9296, 2019. 2

[3] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 5, 7

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 1, 2

[5] Li Chen, Chonghao Sima, Yang Li, Zehan Zheng, Jiajie Xu, Xiangwei Geng, Hongyang Li, Conghui He, Jianping Shi, Yu Qiao, et al. Persformer: 3d lane detection via perspective transformer and the openlane benchmark. *arXiv preprint arXiv:2203.11089*, 2022. 2, 4, 5, 6, 7

[6] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2147–2156, 2016. 2

[7] Noa Garnett, Rafi Cohen, Tomer Pe'er, Roee Lahav, and Dan Levi. 3d-lanenet: end-to-end 3d multiple lane detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2921–2930, 2019. 2, 7

[8] Yuliang Guo, Guang Chen, Peitao Zhao, Weide Zhang, Jinghao Miao, Jingao Wang, and Tae Eun Choe. Gen-lanenet: A generalized and scalable approach for 3d lane detection. In *European Conference on Computer Vision*, pages 666–681. Springer, 2020. 2, 7

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[10] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15273–15282, 2021. 7

[11] Junjie Huang and Guan Huang. Bevdet4d: Exploit temporal cues in multi-camera 3d object detection. *arXiv preprint arXiv:/2203.17054*, 2021. 1, 2, 5, 6, 7

[12] 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. 1, 2, 6, 7

[13] Eskil Jörgensen, Christopher Zach, and Fredrik Kahl. Monocular 3d object detection and box fitting trained end-to-end using intersection-over-union loss. *arXiv preprint arXiv:1906.08070*, 2019. 2

[14] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *Proceedings of the IEEE international conference on computer vision*, pages 1521–1529, 2017. 2

[15] Jason Ku, Alex D Pon, and Steven L Waslander. Monocular 3d object detection leveraging accurate proposals and shape reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11867–11876, 2019. 2

[16] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955. 1, 6

[17] Youngwan Lee and Jongyoul Park. Centermask: Real-time anchor-free instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13906–13915, 2020. 5, 6, 7

[18] Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M Ni, and Lei Zhang. Dn-detr: Accelerate detr training by introducing query denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13619–13627, 2022. 6

[19] Qi Li, Yue Wang, Yilun Wang, and Hang Zhao. Hdmapnet: A local semantic map learning and evaluation framework. *arXiv preprint arXiv:2107.06307*, 2021. 2

[20] 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 arXiv:2203.17270*, 2022. 1, 2, 5, 6, 7

[21] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 6

[22] Lizhe Liu, Xiaohao Chen, Siyu Zhu, and Ping Tan. Condlanenet: a top-to-down lane detection framework based on conditional convolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3773–3782, 2021. 7

[23] Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. Dab-detr: Dynamic anchor boxes are better queries for detr. *arXiv preprint arXiv:2201.12329*, 2022. 2

[24] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petr: Position embedding transformation for multi-view 3d object detection. *arXiv preprint arXiv:2203.05625*, 2022. 1, 2, 3, 4, 5, 6, 7

[25] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 6

[26] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 6

[27] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

[28] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3651–3660, 2021. 2

[29] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7074–7082, 2017. 2

[30] 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*, 5(3):4867–4873, 2020. 2

[31] Dennis Park, Rares Ambrus, Vitor Guizilini, Jie Li, and Adrien Gaidon. Is pseudo-lidar needed for monocular 3d object detection? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3142–3152, 2021. 7

[32] Lang Peng, Zhirong Chen, Zhangjie Fu, Pengpeng Liang, and Erkang Cheng. Bevsegformer: Bird's eye view semantic segmentation from arbitrary camera rigs. *arXiv preprint arXiv:2203.04050*, 2022. 2

[33] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *European Conference on Computer Vision*, pages 194–210. Springer, 2020. 1, 2, 5, 6, 7

[34] Danila Rukhovich, Anna Vorontsova, and Anton Konushin. Imvoxelnet: Image to voxels projection for monocular and multi-view general-purpose 3d object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2397–2406, 2022. 2

[35] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1991–1999, 2019. 2

[36] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 5, 7

[37] Tai Wang, ZHU Xinge, Jiangmiao Pang, and Dahua Lin. Probabilistic and geometric depth: Detecting objects in perspective. In *Conference on Robot Learning*, pages 1475–1485. PMLR, 2022. 2, 6, 7

[38] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Fcos3d: Fully convolutional one-stage monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 913–922, 2021. 2, 6, 7

[39] Yue Wang, Guizilini Vitor Campagnolo, Tianyuan Zhang, Hang Zhao, and Justin Solomon. Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In *In Conference on Robot Learning*, pages 180–191, 2022. 1, 2, 6, 7

[40] Enze Xie, Zhiding Yu, Daquan Zhou, Jonah Philion, Anima Anandkumar, Sanja Fidler, Ping Luo, and Jose M Alvarez. Mˆ 2bev: Multi-camera joint 3d detection and segmentation with unified birds-eye view representation. *arXiv preprint arXiv:2204.05088*, 2022. 2, 7

[41] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 7

[42] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11784–11793, 2021. 2

[43] Brady Zhou and Philipp Krähenbühl. Cross-view transformers for real-time map-view semantic segmentation. *arXiv preprint arXiv:2205.02833*, 2022. 2, 4

[44] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 6, 7

[45] 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

[46] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 2