Tri-Perspective View for Vision-Based 3D Semantic Occupancy Prediction

Yuanhui Huang*   Wenzhao Zheng*   Yunpeng Zhang   Jie Zhou   Jiwen Lu†

Beijing National Research Center for Information Science and Technology, China

Department of Automation, Tsinghua University, China

{huangyh22,zhengwz18}@mails.tsinghua.edu.cn; yunpengzhang97@gmail.com;
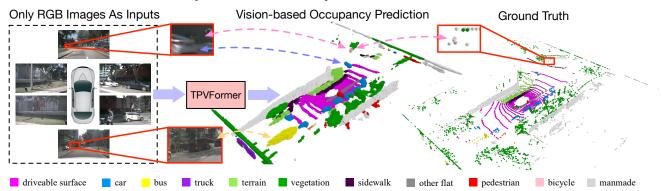
{jzhou,lujiwen}@tsinghua.edu.cn

Figure 1. Given only surround-camera RGB images as inputs, our model (trained using only sparse LiDAR point supervision) can predict the semantic occupancy for all volumes in the 3D space. This task is challenging as it requires both geometric and semantic understandings of the 3D scene. We observe that our model can produce even more comprehensive and consistent volume occupancy than the groundtruth on the validation set (not seen during training) of nuScenes [4]. Despite the lack of geometric inputs like LiDAR, our model can accurately identify the 3D positions and sizes of close and distant objects. Particularly, our model even successfully identifies the partially occluded bicycle captured only by two LiDAR points, demonstrating the potential advantage of vision-based 3D semantic occupancy prediction.

Abstract

Modern methods for vision-centric autonomous driving perception widely adopt the bird's-eye-view (BEV) representation to describe a 3D scene. Despite its better efficiency than voxel representation, it has difficulty describing the fine-grained 3D structure of a scene with a single plane. To address this, we propose a tri-perspective view (TPV) representation which accompanies BEV with two additional perpendicular planes. We model each point in the 3D space by summing its projected features on the three planes. To lift image features to the 3D TPV space, we further propose a transformer-based TPV encoder (TPVFormer) to obtain the TPV features effectively. We employ the attention mechanism to aggregate the image features corresponding to each query in each TPV plane. Experiments show that our model trained with sparse supervision effectively predicts the semantic occupancy for all voxels. We demonstrate for the first time that using only camera inputs can achieve comparable performance with LiDAR-based methods on the LiDAR segmentation task on nuScenes. Code: https://github.com/wzzheng/TPVFormer.

1. Introduction

Perceiving the 3D surroundings accurately and comprehensively plays an important role in the autonomous driving system. Vision-based 3D perception recently emerges as a promising alternative to LiDAR-based one to effectively extract 3D information from 2D images. Though lacking direct sensing of depth information, vision-based models empowered by surrounding cameras demonstrate promising performance on various 3D perception tasks such as depth estimation [17, 42], semantic map reconstruction [1, 19, 48], and 3D object detection [27, 30, 46].

The core of 3D surrounding perceiving lies in how to effectively represent a 3D scene. Conventional methods split the 3D space into voxels and assign each voxel a vector to represent its status. Despite its accuracy, the vast number of voxels poses a great challenge to computation and requires specialized techniques like sparse convolution [13]. As the information in outdoor scenes is not isotropically distributed, modern methods collapse the height dimension and mainly focus on the ground plane (bird's-eye-view) where information varies the most [20, 26, 28, 31, 35, 46, 48].

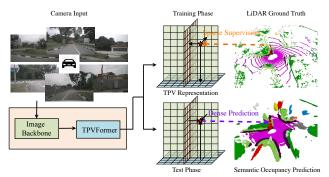*Equal contribution. †Corresponding author.

Figure 2. An overview of our method for 3D semantic occupancy prediction. Taking camera images as inputs, the proposed TPV-Former only uses sparse LiDAR semantic labels for training but can effectively predict the semantic occupancy for all voxels.

They implicitly encode the 3D information of each object in the vector representation in each BEV grid. Though more efficient, BEV-based methods perform surprisingly well on the 3D object detection task [28, 31]. This is because 3D object detection only demands predictions of coarse-level bounding boxes for commonly seen objects such as cars and pedestrians. However, objects with various 3D structures can be encountered in real scenes and it is difficult (if not impossible) to encode all of them using a flattened vector. Therefore, it requires a more comprehensive and fine-grained understanding of the 3D surroundings toward a safer and more robust vision-centric autonomous driving system. Still, it remains unknown how to generalize BEV to model fine-grained 3D structures while preserving its efficiency and detection performance.

In this paper, we advance in this direction and propose a tri-perspective view (TPV) representation to describe a 3D scene. Motivated by recent advances in explicit-implicit hybrid scene representations [7, 8], we generalize BEV by accompanying it with two perpendicular planes to construct three cross-planes perpendicular to each other. Each plane models the 3D surroundings from one view and combining them provides a comprehensive description of the 3D structure. Specifically, to obtain the feature of a point in the 3D space, we first project it into each of the three planes and use bilinear interpolation to obtain the feature for each projected point. We then sum the three projected features as the comprehensive feature of the 3D point. The TPV representation is thus able to describe the 3D scene at an arbitrary resolution and produces different features for different points in the 3D space. We further propose a transformer-based encoder (TPVFormer) to effectively obtain the TPV features from 2D images. We first perform image cross-attention between TPV grid queries and the corresponding 2D image features to lift 2D information to the 3D space. We then perform cross-view hybrid-attention among the TPV features to enable interactions among the three planes.

To demonstrate the superiority of TPV, we formulate a practical yet challenging task for vision-based 3D semantic

occupancy prediction, where only sparse lidar semantic labels are provided for training and predictions for all voxels are required for testing, as shown in Figure 2. However, as no benchmark is provided on this challenging setting, we only perform qualitative analysis but provide a quantitative evaluation on two proxy tasks: LiDAR segmentation (sparse training, sparse testing) on nuScenes [4] and 3D semantic scene completion (dense training, dense testing) on SemanticKITTI [2]. For both tasks, we only use RGB images as inputs. For LiDAR segmentation, our model use the LiDAR data only for point query to compute evaluation metrics. Visualization results show that TPVFormer produces consistent semantic voxel occupancy prediction with only sparse point supervision during training, as shown in Figure 1. We also demonstrate for the first time that our vision-based method achieves comparable performance with LiDAR-based methods on LiDAR segmentation.

2. Related Work

Voxel-based Scene Representation: Obtaining an effective representation for a 3D scene is the basic procedure for 3D surrounding perception. One direct way is to discretize the 3D space into voxels and assign a vector to represent each voxel [49, 51]. The ability to describe fine-grained 3D structures makes voxel-based representation favorable for 3D semantic occupancy prediction tasks including lidar segmentation [12, 29, 40, 44, 45, 51] and 3D scene completion [5, 10, 23, 38, 43]. Though they have dominated the 3D segmentation task [44], they still lag behind BEV-based methods on the 3D detection performance [26]. Despite the success of voxel-based representations in LiDAR-centric surrounding perception, only a few works have explored voxel-based representations for vision-centric autonomous driving [5,25]. MonoScene [5] first backprojects image features to all possible positions in the 3D space along the optical ray to obtain the initial voxel representation and further processes it using a 3D UNet. However, it is still challenging to generalize it to 3D perception with multi-view images due to the inefficiency of voxel representations. This motivates us to explore more efficient and expressive ways to describe the fine-grained 3D structure of a scene.

BEV-based Scene Representation: The vast number of voxels poses a great challenge to the computation efficiency of voxel-based methods. Considering that the height dimension contains less information than the other two dimensions, BEV-based methods implicitly encode the height information in each BEV grid for a more compact representation of scenes [22]. Recent studies in BEV-based perception focus on how to effectively transform features from the image space to the BEV space [20, 26, 27, 35, 36, 48]. One line of works explicitly predict a depth map for each image and utilizes it to project image features into the 3D space followed by BEV pooling [20, 26, 28, 31, 35, 36, 48].

Another line of works employ BEV queries to implicitly assimilate information from image features using the cross-attention mechanism [21, 27]. BEV-based perception achieves great success on vision-centric 3D detection from multi-view images [26], demonstrating comparable performance to LiDAR-centric methods. Yet, it is difficult to apply BEV to 3D semantic occupancy prediction which requires a more fine-grained description of the 3D space.

Implicit Scene Representation: Recent methods have also explored implicit representations to describe a scene. They learn a continuous function that takes as input the 3D coordinate of a point and outputs the representation of this point [32–34]. Compared with explicit representations like voxel and BEV, implicit representations usually share the advantage of arbitrary-resolution modeling and computation-efficient architectures [6, 11, 37]. These advantages enable them to scale to larger and more complex scenes with more fine-grained descriptions. Especially, our work is inspired by recent advances in hybrid explicit-implicit representations [7, 8]. They explicitly inject spatial information into the continuous mapping of implicit representations. Therefore, they share the computation-efficient architecture of implicit representations and better spatial awareness of explicit representations. Still, they mainly focus on small-scale complex scenes for 3D-aware image rendering. To the best of our knowledge, we are the first to use implicit representation to model outdoor scenes for 3D surrounding perception in autonomous driving.

3. Proposed Approach

3.1. Generalizing BEV to TPV

Autonomous driving perception requires both expressive and efficient representation of the complex 3D scene. Voxel representation [25, 40, 45] describes a 3D scene with dense cubic features $\mathbf{V} \in \mathbb{R}^{H \times W \times D \times C}$ where H, W, D are the spatial resolution of the voxel space and C denotes the feature dimension. A random point located at (x, y, z) in the real world maps to its voxel coordinates (h, w, d) through one-to-one correspondence. Therefore, voxel representation preserves the dimensionality of the real world and offers sufficient expressiveness with appropriate H, W, D. However, the storage and computation complexity of voxel features comes proportion to $O(HWD)$, making it challenging to deploy them in real-time onboard applications.

As a popular alternative, BEV [21, 26, 27, 31] representation uses a 2D feature map $\mathbf{B} \in \mathbb{R}^{H \times W \times C}$ to encode the top view of a scene. Different from the voxel counterpart, the point at (x, y, z) is projected to its BEV coordinates (h, w) using only the positional information from the ground plane regardless of the z-axis. Each feature sampled from $\mathbf{B}$ corresponds to a pillar region covering the full range of z-axis in the real world. Although BEV greatly reduces the storage
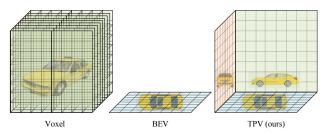

Figure 3. Comparisons of the proposed TPV representation with voxel and BEV representation. While BEV is more efficient than the voxel representation, it discards the height information and cannot comprehensively describe a 3D scene.

and computation burden to $O(HW)$, completely omitting the z-axis has an adverse effect on its expressiveness.

To address this, we propose a Tri-Perspective View (TPV) representation which is capable of modeling the 3D space at full scale without suppressing any axes and avoiding cubic complexity, as illustrated in Figure 3. Formally, we learn three axis-aligned orthogonal TPV planes:

$$\mathbf{T} = [\mathbf{T}^{HW}, \mathbf{T}^{DH}, \mathbf{T}^{WD}], \quad \mathbf{T}^{HW} \in \mathbb{R}^{H \times W \times C},$$
$$\mathbf{T}^{DH} \in \mathbb{R}^{D \times H \times C}, \quad \mathbf{T}^{WD} \in \mathbb{R}^{W \times D \times C}, \tag{1}$$

which represent the top, side and front views of a 3D scene respectively. H, W, D denote the resolution of the three planes and C is the feature dimension. Intuitively, a complex scene, when examined from different perspectives, can be better understood because these perspectives may provide complementary clues about the scene.

Point Querying Formulation. Given a query point at (x, y, z) in the real world, TPV representation aggregates its projections on the top, side and front views in order to get a comprehensive description of the point. To elaborate, we first project the point onto the TPV planes to obtain the coordinates $[(h, w), (d, h), (w, d)]$, sample the TPV planes at these locations to retrieve the corresponding features $[\mathbf{t}_{h,w}, \mathbf{t}_{d,h}, \mathbf{t}_{w,d}]$, and aggregate them to generate the final $\mathbf{f}_{x,y,z}$:

$$\mathbf{t}_{i,j} = \mathcal{S}(\mathbf{T}, (i, j)) = \mathcal{S}(\mathbf{T}, \mathcal{P}(x, y)), \tag{2}$$

$$\mathbf{f}_{x,y,z} = \mathcal{A}(\mathbf{t}_{h,w}, \mathbf{t}_{d,h}, \mathbf{t}_{w,d}), \tag{3}$$

where the sampling function $\mathcal{S}$ and the aggregation function $\mathcal{A}$ are implemented with bilinear interpolation and summation respectively, and each projection function $\mathcal{P}$ simply performs scaling on the two relevant coordinates since TPV planes are aligned with the real-world axes.

Voxel Feature Formulation. The TPV planes, when repeated along respective orthogonal directions and summed up, construct a full-scale 3D feature space similar to the voxel feature space, but with storage and computation complexity of only $O(HW + DH + WD)$, which is an order of magnitude lower than the voxel counterpart.

Compared with BEV, as the three planes in TPV are perpendicular to each other, point features along the orthogonal direction of one plane are diversified by features sampled from the other two planes. Moreover, a grid feature in each TPV plane is only responsible for view-specific information of the corresponding pillar region rather than encoding the complete information as in BEV. To sum up, TPV representation generalizes BEV from single top view to complementary and orthogonal top, side and front views and is able to offer a more comprehensive and fine-grained understanding of the 3D surroundings while remaining efficient.

3.2. TPVFormer

For vision-centric autonomous driving perception, a 2D backbone is often employed to obtain image features before feeding them into a 3D encoder. We present a transformer-based TPV encoder (TPVFormer) to lift image features to TPV planes through the attention mechanism.

Overall Structure: In TPVFormer, we introduce TPV queries, image cross-attention (ICA) and cross-view hybrid-attention (CVHA) to enable effective generation of TPV planes, as shown in Fig. 4. In fact, TPV queries and TPV planes refer to the same set of feature vectors defined in (1). Each TPV query $\mathbf{t} \in \mathbf{T}$ is a grid cell feature belonging to one of the three planes and used to encode view-specific information from the corresponding pillar region. Cross-view hybrid-attention enables direct interactions among TPV queries from the same or different tpv planes in order to gather contextual information. Inside image cross-attention, TPV queries aggregate visual information from image features through deformable attention.

We further construct two kinds of transformer blocks: hybrid-cross-attention block (HCAB) and hybrid-attention block (HAB). Composed of both CVHA and ICA attention, the HCAB block is employed in the first half of TPV-Former to effectively query visual information from image features. Following HCAB blocks, the HAB block contains only CVHA attention and specializes in contextual information encoding. Finally, we build TPVFormer by stacking N_1 HCAB blocks and N_2 HAB blocks.

TPV Queries: Although referring to the same list of 2D features defined in (1), TPV queries and TPV planes are used in attention and 3D representation contexts, respectively. Each TPV query maps to a 2D grid cell region of size $s \times s \ m^2$ in the corresponding view, and further to a 3D pillar region extending from the view in the perpendicular direction. In our pipeline, TPV queries are first enhanced with raw visual information from image features in HCAB blocks, and then refined with contextual clues from other queries in HAB blocks. As for implementation, we initialize TPV queries as learnable parameters.

Image Cross-Attention: In TPVFormer, we use image cross-attention to lift multi-scale and possibly multi-camera

image features to the TPV planes. Considering the high resolution nature of TPV queries ($\sim 10^4$ queries) and multiple image feature maps ($\sim 10^5$ pixels each), it is unfeasible to compute full-scale vanilla cross-attention between them. And thus we employ the efficient deformable attention [14, 27, 50] to implement image cross-attention.

We take the local receptive field as an inductive bias when sampling the reference points. Specifically, for a TPV query $\mathbf{t}_{h,w}$ located at (h, w) in the top plane, we first calculate its coordinates (x, y) in the top view in the real world through the inverse projection function $\mathcal{P}_{HW}^{-1}$. Then we sample uniformly N_{HW}^{ref} reference points for the query $\mathbf{t}_{h,w}$ along the orthogonal direction of the plane:

$$(x, y) = \mathcal{P}_{HW}^{-1}(h, w) = ((h - \frac{H}{2}) \times s, (w - \frac{W}{2}) \times s). \quad (4)$$

$$\mathbf{Ref}_{h,w}^{w} = \{(x, y, z_i)\}_{i=1}^{N_{HW}^{ref}}, \quad (5)$$

where $\mathbf{Ref}_{h,w}^{w}$ denotes the set of reference points in the world coordinate for query $\mathbf{t}_{h,w}$. The similar procedure is repeated for all TPV queries, and note that the number of reference points N^{ref} may change across planes because of different ranges of axes. After deriving the reference points for $\mathbf{t}_{h,w}$, we need to project them into the pixel coordinate in order to sample the image feature maps:

$$\mathbf{Ref}_{h,w}^{p} = \mathcal{P}_{pix}(\mathbf{Ref}_{h,w}^{w}) = \mathcal{P}_{pix}(\{(x, y, z_i)\}), \quad (6)$$

where $\mathbf{Ref}_{h,w}^{p}$ is the set of reference points in the pixel coordinate for query $\mathbf{t}_{h,w}$ and $\mathcal{P}_{pix}$ is the perspective projection function determined by the camera extrinsic and intrinsic. Note that we may have multiple cameras in different directions which will generate a set of $\{\mathbf{Ref}_{h,w}^{p,j}\}_{j=1}^{N_c}$ where N_c denotes the number of cameras. Since not all cameras can capture the reference points of query $\mathbf{t}_{h,w}$, we can further reduce computation by removing invalid sets from $\{\mathbf{Ref}_{h,w}^{p,j}\}_{j=1}^{N_c}$ if none of the reference points falls onto the image captured by the corresponding camera. The final step is to generate offsets and attention weights through two linear layers applied on $\mathbf{t}_{h,w}$ and produce the updated TPV queries by summing up the sampled image features weighted by their attention weights:

$$\text{ICA}(\mathbf{t}_{h,w}, \mathbf{I}) = \frac{1}{|N_{h,w}^{val}|} \sum_{j \in N_{h,w}^{val}} \text{DA}(\mathbf{t}_{h,w}, \mathbf{Ref}_{h,w}^{p,j}, \mathbf{I}_j), \quad (7)$$

where $N_{h,w}^{val}$, $\mathbf{I}_j$, $\text{DA}(\cdot)$ denote the index set of valid cameras, the image features from the jth camera and the deformable attention function, respectively.

Cross-View Hybrid-Attention: In image cross-attention, TPV queries sample reference image features separately and no direct interactions between them are allowed. Here, we further propose cross-view hybrid-attention to enable queries to exchange their information across different
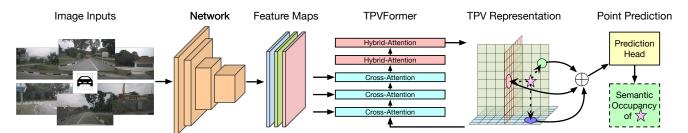
Figure 4. Framework of the proposed TPVFormer for 3D semantic occupancy prediction. We employ an image backbone network to extract multi-scale features for multi-camera images. We then perform cross-attention to adaptively lift 2D features to the TPV space and use cross-view hybrid attention to enable the interactions between TPV planes. To predict the semantic occupancy of a point in the 3D space, we apply a lightweight prediction head on the sum of projected features on the three TPV planes.

views, which benefits context extraction. We also adopt deformable attention to reduce computation, in which three TPV planes serve as key and value. Taking the TPV query $\mathbf{t}_{h,w}$ located at (h, w) in the top plane as an example, we group its reference points into three disjoint subsets, which contains reference points belonging to the top, side and front planes respectively:

$$\mathbf{R}_{h,w} = \mathbf{R}_{h,w}^{top} \cup \mathbf{R}_{h,w}^{side} \cup \mathbf{R}_{h,w}^{front}. \qquad (8)$$

To collect reference points on the top plane, we simply sample a few random points in the neighborhood of the query $\mathbf{t}_{h,w}$. As for the side and front planes, we first sample 3D points uniformly along the direction perpendicular to the top plane and project them onto the side and front planes:

$$\mathbf{R}_{h,w}^{side} = \{(d_i, h)\}_i, \quad \mathbf{R}_{h,w}^{front} = \{(w, d_i)\}_i. \qquad (9)$$

Following the derivation of reference points is the typical practice of deformable attention:

$$\mathrm{CVHA}(\mathbf{t}_{h,w}) = \mathrm{DA}(\mathbf{t}_{h,w}, \mathbf{R}_{h,w}, \mathbf{T}). \qquad (10)$$

3.3. Applications of TPV

The TPV planes $\mathbf{T}$ from TPVFormer encode fine-grained view-specific information of a 3D scene. Still, they are in the form of orthogonal cross-planes and not readily interpretable to common task heads. Here we explain how to convert TPV planes to point and voxel features and introduce a lightweight segmentation head.

Point Feature. Given locations in the real world, we consider the feature generation process as the points querying their features from the TPV representation. As defined in (2) and (3), we first project the points onto the TPV planes to retrieve the corresponding features $[\mathbf{t}_{h,w}, \mathbf{t}_{d,h}, \mathbf{t}_{w,d}]$, and sum them up to obtain the per-point features.

Voxel Feature. For dense voxel features, we actively broadcast each TPV plane along the corresponding orthogonal direction to produce three feature tensors of the same size $H \times W \times D \times C$, and aggregate them by summation to obtain the full-scale voxel features. Note that we do not know the position of any physical point in advance.

To conduct fine-grained segmentation tasks, we apply a lightweight MLP on the point or voxel features to predict their semantic labels, instantiated by only two linear layers and an intermediate activation layer.

4. Experiments

4.1. Task Descriptions

We conduct three types of experiments, including 3D semantic occupancy prediction, LiDAR segmentation, and semantic scene completion (SSC). The first two tasks are performed on Panoptic nuScenes [16], and the last one is on Semantic KITTI [2]. For all tasks, our model only uses RGB images as inputs.

3D semantic occupancy prediction. As dense semantic labels are difficult to obtain, we formulate a practical yet challenging task for vision-based 3D semantic occupancy prediction. Under this task, the model is only trained using sparse semantic labels (LiDAR points) but is required to produce a semantic occupancy for all the voxels in the concerned 3D space during testing. As no benchmark is provided for this, we only perform a qualitative analysis of our method. Still, our method is the first to demonstrate effective results on this challenging task.

LiDAR segmentation. The LiDAR segmentation task corresponds to the point querying formulation discussed in Section 3.3, where we predict the semantic label of a given point. The LiDAR segmentation task does not necessarily use point clouds as input. In our case, we use only RGB images as input, while the points are merely used to query their features and for supervision in the training phase.

Semantic Scene Completion. In conventional SSC, given a single initial LiDAR scan, one needs to predict whether each voxel is occupied and its semantic label for the complete scene inside a certain volume. As a vision-centric adaptation, we use as input only RGB images and predict the occupancy and semantic label of each voxel. Accordingly, we supervise the training process with voxel labels. In the case of TPV representation, we adopt the voxel feature formulation in Section 3.3 to generate full-scale voxel features. Following common practices, we report the inter-
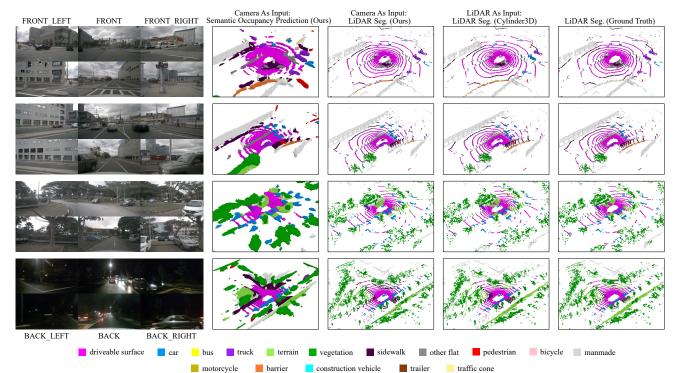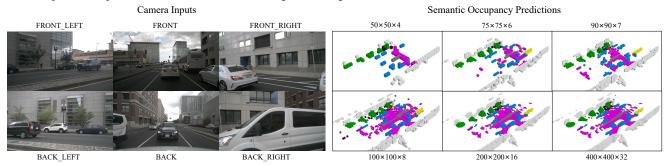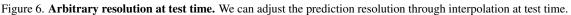
Figure 5. **Visualization results on 3D semantic occupancy prediction and nuScenes LiDAR segmentation.** Our method can generate more comprehensive prediction results than the LiDAR segmentation ground truth.

Figure 6. **Arbitrary resolution at test time.** We can adjust the prediction resolution through interpolation at test time.

section over union (IoU) of occupied voxels, ignoring their semantic class, for the scene completion (SC) task and the mIoU of all semantic classes for the SSC task.

4.2. Implementation Details

3D semantic occupancy prediction and LiDAR segmentation. We construct two versions of TPVFormer, namely TPVFormer-Base and TPVFormer-Small, for different trade-offs between performance and efficiency. TPVFormer-Base uses the ResNet101-DCN [14, 18] initialized from FCOS3D [41] checkpoint, while TPVFormer-Small adopts the ResNet-50 [18] pretrained on ImageNet [15]. Following Cylinder3D [51], we employ both cross entropy loss and lovasz-softmax [3] loss to optimize our network. For lovasz-softmax loss, we use features of real points from LiDAR scans as input to maximize the IoU score for classes, while voxel features are used in cross en-

tropy loss to improve point classification accuracy and avoid semantic ambiguity. For 3D semantic occupancy prediction, we generate pseudo-per-voxel labels from sparse point cloud by assigning a new label of empty to any voxel that does not contain any point, and we use voxel predictions as input to both lovasz-softmax and cross-entropy losses.

Semantic Scene Completion. We follow the setting of MonoScene [5] in the SSC task for a fair comparison. For model architecture, we adopt the 2D UNet based on a pretrained EfficientNetB7 [39] as image backbone to generate multi-scale image features. For optimization, we employ the losses in MonoScene except for the relation loss.

4.3. 3D Semantic Occupancy Prediction Results

Main results. In Figure 5, we provide the main visualization results. Our result is much denser and more realistic than the LiDAR segmentation ground truth, which validates

Table 1. **LiDAR segmentation results on nuScenes test set.** Despite critical modal difference, our TPVFormer-Base achieves comparable performance with LiDAR-based methods.

Method	Input Modality	mIoU	barrier	bicycle	bus	car	const. veh.	motorcycle	pedestrian	traffic cone	trailer	truck	drive. suf.	other flat	sidewalk	terrain	manmade	vegetation
MINet [24]	LiDAR	56.3	54.6	8.2	62.1	76.6	23.0	58.7	37.6	34.9	61.5	46.9	93.3	56.4	63.8	64.8	79.3	78.3
PolarNet [47]	LiDAR	69.4	72.2	16.8	77.0	86.5	51.1	69.7	64.8	54.1	69.7	63.5	96.6	67.1	77.7	72.1	87.1	84.5
PolarSteam [9]	LiDAR	73.4	71.4	27.8	78.1	82.0	61.3	77.8	75.1	72.4	79.6	63.7	96.0	66.5	76.9	73.0	88.5	84.8
JS3C-Net [43]	LiDAR	73.6	80.1	26.2	87.8	84.5	55.2	72.6	71.3	66.3	76.8	71.2	96.8	64.5	76.9	74.1	87.5	86.1
AMVNet [29]	LiDAR	77.3	80.6	32.0	81.7	88.9	67.1	84.3	76.1	73.5	84.9	67.3	97.5	67.4	79.4	75.5	91.5	88.7
SPVNAS [40]	LiDAR	77.4	80.0	30.0	91.9	90.8	64.7	79.0	75.6	70.9	81.0	74.6	97.4	69.2	80.0	76.1	89.3	87.1
Cylinder3D++ [51]	LiDAR	77.9	82.8	33.9	84.3	89.4	69.6	79.4	77.3	73.4	84.6	69.4	97.7	70.2	80.3	75.5	90.4	87.6
AF2S3Net [12]	LiDAR	78.3	78.9	**52.2**	89.9	84.2	**77.4**	74.3	77.3	72.0	83.9	73.8	97.1	66.5	77.5	74.0	87.7	86.8
DRINet++ [45]	LiDAR	80.4	**85.5**	43.2	90.5	**92.1**	64.7	86.0	83.0	73.3	83.9	**75.8**	97.0	**71.0**	**81.0**	77.7	91.6	**90.2**
LidarMultiNet [44]	LiDAR	**81.4**	80.4	48.4	**94.3**	90.0	71.5	**87.2**	**85.2**	80.4	**86.9**	74.8	**97.8**	67.3	80.7	76.5	**92.1**	89.6
TPVFormer-Small (ours)	Camera	59.2	65.6	15.7	75.1	80.0	45.8	43.1	44.3	26.8	72.8	55.9	92.3	53.7	61.0	59.2	79.7	75.6
TPVFormer-Base (ours)	Camera	69.4	74.0	27.5	86.3	85.5	60.7	68.0	62.1	49.1	81.9	68.4	94.1	59.5	66.5	63.5	83.8	79.9

Table 2. **Semantic scene completion results on SemanticKITTI test set.** For fair comparison, we use the performances of RGB-inferred versions of the first four methods reported in MonoScene [5]. We significantly outperform other methods in both IoU and mIoU, including MonoScene which is based on 3D convolution.

Method	Input Modality	SC IoU	SSC mIoU	road (15.30%)	sidewalk (11.13%)	parking (1.12%)	other-grnd (0.56%)	building (14.1%)	car (3.92%)	truck (0.16%)	bicycle (0.03%)	motorcycle (0.03%)	other-veh. (0.20%)	vegetation (39.3%)	trunk (0.51%)	terrain (9.17%)	person (0.07%)	bicyclist (0.07%)	motorcyclist. (0.05%)	fence (3.90%)	pole (0.29%)	traf.-sign (0.08%)
LMSCNet [38]	Camera	31.38	7.07	46.70	19.50	13.50	3.10	10.30	14.30	0.30	0.00	0.00	0.00	10.80	0.00	10.40	0.00	0.00	0.00	5.40	0.00	0.00
3DSketch [10]	Camera	26.85	6.23	37.70	19.80	0.00	0.00	12.10	17.10	0.00	0.00	0.00	0.00	12.10	0.00	16.10	0.00	0.00	0.00	3.40	0.00	0.00
AICNet [23]	Camera	23.93	7.09	39.30	18.30	19.80	1.60	9.60	15.30	0.70	0.00	0.00	0.00	9.60	1.90	13.50	0.00	0.00	0.00	5.00	0.10	0.00
JS3C-Net [43]	Camera	34.00	8.97	47.30	21.70	19.90	2.80	12.70	**20.10**	0.80	0.00	0.00	4.10	14.20	3.10	12.40	0.00	0.20	0.20	8.70	1.90	0.30
MonoScene [5]	Camera	34.16	11.08	54.70	27.10	24.80	5.70	14.40	18.80	3.30	0.50	0.70	4.40	14.90	2.40	19.50	1.00	1.40	0.40	11.10	3.30	2.10
TPVFormer (ours)	Camera	34.25	11.26	55.10	27.20	27.40	6.50	14.80	19.20	3.70	1.00	0.50	2.30	13.90	2.60	20.40	1.10	2.40	0.30	11.00	2.90	1.50

the effectiveness of TPV representation in modeling the 3D scene and semantic occupancy prediction.

Arbitrary resolution at test time. Given the simplicity of our segmentation head, we can adjust the resolution of TPV planes at test time arbitrarily without retraining the network. Figure 6 shows the results for resolution adjustment, in which we gradually increase the resolution of TPV planes from an initial 50x50x4 to 8 times larger. It is evident that as resolution increases, TPV representation is able to capture more details about the 3D objects, such as shape.

4.4. LiDAR Segmentation Results

As the first vision-based method for LiDAR segmentation task, we benchmark TPVFormer against LiDAR-based methods. As shown in Table 1, TPVFormer achieves comparable mIoU ($\sim 70\%$) with most LiDAR-based methods. This is nontrivial since our method needs to reconstruct the complete 3D scene at a high resolution from only 2D image input, while the 3D structural information is readily available in the point clouds for LiDAR-based methods. Still, visual input can capture more semantic information than point clouds. For example, in Figure 5, Cylinder3D fails to predict one of the two trucks on the rightmost side of the first

scene, while TPVFormer predicts correctly.

4.5. Semantic Scene Completion Results

In Table 2, we report the results of the semantic scene completion task on SemanticKITTI test set. We compare our TPVFormer against MonoScene [5], which is a vision-based method based on 3D convolution in the voxel space. We also include the 4 baseline methods provided in MonoScene [5]. TPVFormer outperforms all other methods in both IoU and mIoU, demonstrating its effectiveness in occupancy and semantics prediction. TPVFormer also enjoys significant advantages over MonoScene in both parameter number and computation. Specifically, TPVFormer has only 6.0M parameters versus 15.7M for MonoScene, and 128G FLOPS per image versus 500G for MonoScene.

4.6. Analysis

We analyze TPVFormer on the validation sets of nuScenes and SemanticKITTI for LiDAR segmentation and semantic scene completion, respectively.

Loss functions for LiDAR segmentation. We employ both cross entropy (CE.) loss and lovasz-softmax loss [3] for LiDAR segmentation. As TPVFormer can produce

Table 3. **Different prediction types as input to loss functions for LiDAR segmentation.** Voxel and point in the loss column represent voxel and point predictions. We report mIoUs calculated with both voxel and point predictions.

| Loss | | mIoU | |
CE.	Lovasz	Voxel	Point
Voxel	Voxel	63.17	50.66
Voxel	Point	63.37	**64.80**
Point	Voxel	**64.07**	64.46
Point	Point	49.94	64.02

Table 4. **Comparisons between BEV, voxel, TPV representations for LiDAR segmentation.** Latency at inference is measured with a batch size of 1 using a RTX 3090 GPU.

Repr.	Feature	Resolution	#Param. (M)	FLOPs (G)	Latency (s)	mIoU (%)
BEV	256	200x200	61.2	1351	0.323	54.5
Voxel	64	100x100x8	49.8	915	0.314	24.2
TPV	64	200x200x16	48.8	934	0.312	**67.4**

Table 5. **Ablations on TPV resolutions and feature dimensions.**

TPV Resolution	Feature	Point mIoU (%)
100x100x8	256	64.15
200x200x16	64	**67.40**

point and voxel predictions in a single forward propagation, we investigate different prediction types as input to these loss functions. As shown in Table 3, when both voxel and point predictions are used as input to the loss functions, the mIoUs from both predictions are high and close to each other. However, when only voxel or point prediction is employed in optimization, the corresponding mIoU will be much higher than the other one. This indicates the importance of discrete and continuous supervision signals for learning a robust representation.

TPV resolution and feature dimension. We ablate resolution of TPV and feature dimension for LiDAR segmentation in Table 5. TPVFormer favors resolution over feature dimension because increasing the resolution directly enhances its ability to model more fine-grained structures.

Comparisons between BEV, voxel and TPV. We adopt voxel, BEV and TPV as the representation for 3D space and use a similar architecture to the proposed TPVFormer to lift image features to 3D. We show the comparisons in Table 4. We increase the feature dimension for BEV considering the capacity of the single top view, which accounts for the extra parameters and FLOPs. Since voxel representation takes up much more GPU memory, we have to reduce its grid resolution to 100x100x8. We observe that TPVFormer achieves the highest mIoU with fewer parameters and FLOPs compared with BEV and voxel counterparts. For fair comparisons, we keep the model architectures the same, which may result in the inferior performance of voxel representation and has room for improvement. For latency, sequential processing of three TPV planes as in our code may account for

Table 6. **Different number of HCAB blocks and HAB blocks for semantic scene completion.** We keep the total number of attention modules the same in these experiments.

# HCAB	# HAB	SC IoU	SSC mIoU
2	4	35.55	10.49
3	2	35.61	**11.36**
4	0	**35.79**	10.82

Figure 7. Failure cases of pedestrians for semantic occupancy prediction. TPVFormer has trouble distinguishing pedestrians close to each other and could predict a strip area for a distant pedestrian.

the comparable latency of TPV and voxel.

Number of HCAB and HAB blocks. We study the number of HCAB and HAB blocks in Table 6. The IoU improves as HCAB blocks increase, validating the importance of direct visual clues for geometry understanding. However, semantic prediction relies on both visual and contextual information as the highest mIoU is achieved with a moderate number of HCAB and HAB blocks.

Limitations. As shown in Figure 5, the main strength of the camera-based method over LiDAR-based methods is the ability to output dense predictions for 3D space. On the other hand, current TPVFormer still underperforms LiDAR-based methods for lidar segmentation. In Figure 7, we identify some failure cases on semantic occupancy prediction. TPVFormer fails to seperate pedestrians close together and could predict a distant pedestrian as a strip area.

5. Conclusion

In this paper, we have presented a tri-perspective view (TPV) representation, which is able to describe the fine-grained structures of a 3D scene efficiently. To lift image features to the 3D TPV space, we have proposed a TPVFormer model based on the attention mechanism. The visualization results have shown that our TPVFormer produces consistent semantic voxel occupancy prediction with only sparse point supervision during training. We have demonstrated for the first time that our vision-based method achieves comparable performance with LiDAR-based methods on nuScenes LiDAR segmentation task.

Acknowledgement

References

[1] Adil Kaan Akan and Fatma Güney. Stretchbev: Stretching future instance prediction spatially and temporally. *arXiv preprint arXiv:2203.13641*, 2022. 1

[2] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *ICCV*, pages 9297–9307, 2019. 2, 5

[3] Maxim Berman, Amal Rannen Triki, and Matthew B Blaschko. The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In *CVPR*, pages 4413–4421, 2018. 6, 7

[4] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020. 1, 2

[5] Anh-Quan Cao and Raoul de Charette. Monoscene: Monocular 3d semantic scene completion. In *CVPR*, pages 3991–4001, 2022. 2, 6, 7

[6] Rohan Chabra, Jan E Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *ECCV*, pages 608–625, 2020. 3

[7] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *CVPR*, pages 16123–16133, 2022. 2, 3

[8] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. *arXiv preprint arXiv:2203.09517*, 2022. 2, 3

[9] Qi Chen, Sourabh Vora, and Oscar Beijbom. Polarstream: Streaming object detection and segmentation with polar pillars. *NeurIPS*, 34:26871–26883, 2021. 7

[10] Xiaokang Chen, Kwan-Yee Lin, Chen Qian, Gang Zeng, and Hongsheng Li. 3d sketch-aware semantic scene completion via semi-supervised structure prior. In *CVPR*, pages 4193–4202, 2020. 2, 7

[11] Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *CVPR*, pages 8628–8638, 2021. 3

[12] Ran Cheng, Ryan Razani, Ehsan Taghavi, Enxu Li, and Bingbing Liu. 2-s3net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network. In *CVPR*, pages 12547–12556, 2021. 2, 7

[13] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, pages 3075–3084, 2019. 1

[14] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017. 4, 6

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009. 6

[16] Whye Kit Fong, Rohit Mohan, Juana Valeria Hurtado, Lubing Zhou, Holger Caesar, Oscar Beijbom, and Abhinav Valada. Panoptic nuscenes: A large-scale benchmark for lidar panoptic segmentation and tracking. *arXiv preprint arXiv:2109.03805*, 2021. 5

[17] Vitor Guizilini, Igor Vasiljevic, Rares Ambrus, Greg Shakhnarovich, and Adrien Gaidon. Full surround monodepth from multiple cameras. *arXiv preprint arXiv:2104.00152*, 2021. 1

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6

[19] 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*, pages 15273–15282, 2021. 1

[20] 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

[21] Yanqin Jiang, Li Zhang, Zhenwei Miao, Xiatian Zhu, Jin Gao, Weiming Hu, and Yu-Gang Jiang. Polarformer: Multi-camera 3d object detection with polar transformers. *arXiv preprint arXiv:2206.15398*, 2022. 3

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

[23] Jie Li, Kai Han, Peng Wang, Yu Liu, and Xia Yuan. Anisotropic convolutional networks for 3d semantic scene completion. In *CVPR*, pages 3351–3359, 2020. 2, 7

[24] Shijie Li, Xieyuanli Chen, Yun Liu, Dengxin Dai, Cyrill Stachniss, and Juergen Gall. Multi-scale interaction for real-time lidar data segmentation on an embedded platform. *RA-L*, 7(2):738–745, 2021. 7

[25] Yanwei Li, Yilun Chen, Xiaojuan Qi, Zeming Li, Jian Sun, and Jiaya Jia. Unifying voxel-based representation with transformer for 3d object detection. In *Advances in Neural Information Processing Systems*, 2022. 2, 3

[26] Yinhao Li, Zheng Ge, Guanyi Yu, Jinrong Yang, Zengran Wang, Yukang Shi, Jianjian Sun, and Zeming Li. Bevdepth: Acquisition of reliable depth for multi-view 3d object detection. *arXiv preprint arXiv:2206.10092*, 2022. 1, 2, 3

[27] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Qiao Yu, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *ECCV*, 2022. 1, 2, 3, 4

[28] Tingting Liang, Hongwei Xie, Kaicheng Yu, Zhongyu Xia, Zhiwei Lin, Yongtao Wang, Tao Tang, Bing Wang, and Zhi Tang. Bevfusion: A simple and robust lidar-camera fusion framework. *arXiv preprint arXiv:2205.13790*, 2022. 1, 2

[29] Venice Erin Liong, Thi Ngoc Tho Nguyen, Sergi Widjaja, Dhananjai Sharma, and Zhuang Jie Chong. Amvnet: Assertion-based multi-view fusion network for lidar semantic segmentation. *arXiv preprint arXiv:2012.04934*, 2020. 2, 7

[30] 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

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

[32] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, pages 4460–4470, 2019. 3

[33] B Mildenhall, PP Srinivasan, M Tancik, JT Barron, R Ramamoorthi, and R Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 3

[34] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, pages 165–174, 2019. 3

[35] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *ECCV*, pages 194–210, 2020. 1, 2

[36] Cody Reading, Ali Harakeh, Julia Chae, and Steven L Waslander. Categorical depth distribution network for monocular 3d object detection. In *CVPR*, 2021. 2

[37] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *ICCV*, pages 14335–14345, 2021. 3

[38] Luis Roldao, Raoul de Charette, and Anne Verroust-Blondet. Lmscnet: Lightweight multiscale 3d semantic completion. In *2020 International Conference on 3D Vision (3DV)*, pages 111–119. IEEE, 2020. 2, 7

[39] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, pages 6105–6114, 2019. 6

[40] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *ECCV*, pages 685–702, 2020. 2, 3, 7

[41] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Fcos3d: Fully convolutional one-stage monocular 3d object detection. In *ICCV*, 2021. 6

[42] Yi Wei, Linqing Zhao, Wenzhao Zheng, Zheng Zhu, Yongming Rao, Guan Huang, Jiwen Lu, and Jie Zhou. Surround-depth: Entangling surrounding views for self-supervised multi-camera depth estimation. In *CoRL*, 2022. 1

[43] Xu Yan, Jiantao Gao, Jie Li, Ruimao Zhang, Zhen Li, Rui Huang, and Shuguang Cui. Sparse single sweep lidar point cloud segmentation via learning contextual shape priors from scene completion. In *AAAI*, volume 35, pages 3101–3109, 2021. 2, 7

[44] Dongqiangzi Ye, Zixiang Zhou, Weijia Chen, Yufei Xie, Yu Wang, Panqu Wang, and Hassan Foroosh. Lidarmultinet: Towards a unified multi-task network for lidar perception. *arXiv preprint arXiv:2209.09385*, 2022. 2, 7

[45] Maosheng Ye, Rui Wan, Shuangjie Xu, Tongyi Cao, and Qifeng Chen. Drinet++: Efficient voxel-as-point point cloud segmentation. *arXiv preprint arXiv: 2111.08318*, 2021. 2, 3, 7

[46] Yunpeng Zhang, Wenzhao Zheng, Zheng Zhu, Guan Huang, Jie Zhou, and Jiwen Lu. A simple baseline for multi-camera 3d object detection. *arXiv preprint arXiv:2208.10035*, 2022. 1

[47] 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*, pages 9601–9610, 2020. 7

[48] Yunpeng Zhang, Zheng Zhu, Wenzhao Zheng, Junjie Huang, Guan Huang, Jie Zhou, and Jiwen Lu. Beverse: Unified perception and prediction in birds-eye-view for vision-centric autonomous driving. *arXiv preprint arXiv:2205.09743*, 2022. 1, 2

[49] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, pages 4490–4499, 2018. 2

[50] 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. 4

[51] 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*, pages 9939–9948, 2021. 2, 6, 7