PVT-SSD: Single-Stage 3D Object Detector with Point-Voxel Transformer

Honghui Yang[1,3*] Wenxiao Wang[2] Minghao Chen[1] Binbin Lin[2†] Tong He[3†]
Hua Chen[4] Xiaofei He[1] Wanli Ouyang[3]
[1]State Key Lab of CAD&CG, Zhejiang University
[2]School of Software Technology, Zhejiang University
[3]Shanghai AI Laboratory
[4]COMAC Beijing Aircraft Technology Research Institute

Abstract

Recent Transformer-based 3D object detectors learn point cloud features either from point- or voxel-based representations. However, the former requires time-consuming sampling while the latter introduces quantization errors. In this paper, we present a novel Point-Voxel Transformer for single-stage 3D detection (PVT-SSD) that takes advantage of these two representations. Specifically, we first use voxel-based sparse convolutions for efficient feature encoding. Then, we propose a Point-Voxel Transformer (PVT) module that obtains long-range contexts in a cheap manner from voxels while attaining accurate positions from points. The key to associating the two different representations is our introduced input-dependent Query Initialization module, which could efficiently generate reference points and content queries. Then, PVT adaptively fuses long-range contextual and local geometric information around reference points into content queries. Further, to quickly find the neighboring points of reference points, we design the Virtual Range Image module, which generalizes the native range image to multi-sensor and multi-frame. The experiments on several autonomous driving benchmarks verify the effectiveness and efficiency of the proposed method. Code will be available at https://github.com/Nightmare-n/PVT-SSD.

1. Introduction

3D object detection from point clouds has become increasingly popular thanks to its wide applications, e.g., autonomous driving and virtual reality. To process unordered point clouds, Transformer [51] has recently attracted great interest as the self-attention is invariant to the permutation of inputs. However, due to the quadratic complexity of self-attention, it involves extensive computation and memory budgets when processing large point clouds. To overcome this problem, some point-based methods [29, 36, 37] perform attention on downsampled point sets, while some voxel-based methods [10, 33, 64] employ attention on local non-empty voxels. Nevertheless, the former requires farthest point sampling (FPS) [41] to sample point clouds, which is time-consuming on large-scale outdoor scenes [19], while the latter inevitably introduces quantization errors during voxelization, which loses accurate position information.

In this paper, we propose PVT-SSD that absorbs the advantages of the above two representations, i.e., voxels and points, while overcoming their drawbacks. To this end, instead of sampling points directly, we convert points to a small number of voxels through sparse convolutions and sample non-empty voxels to reduce the runtime of FPS. Then, inside the PVT-SSD, voxel features are adaptively fused with point features to make up for the quantization error. In this way, both long-range contexts from voxels and accurate positions from points are preserved. Specifically, PVT-SSD consists of the following components:

Firstly, we propose an input-dependent **Query Initialization** module inspired by previous indoor Transformer-based detectors [29, 36], which provides queries with better initial positions and instance-related features. Unlike [29, 36], our queries originate from non-empty voxels instead of points to reduce the sampling time. Concretely, with the 3D voxels generated by sparse convolutions, we first *collapse* 3D voxels into 2D voxels by merging voxels along the height dimension to further reduce the number of voxels. The *sample* operation is then applied to select a representative set of voxels. We finally *lift* sampled 2D voxels to generate 3D reference points. Subsequently, the corresponding content queries are obtained in an efficient way by projecting reference points onto a BEV feature map and indexing features at the projected locations.

*This work was done when Honghui was an intern at Shanghai Artificial Intelligence Laboratory.
†Corresponding author

Secondly, we introduce a **Point-Voxel Transformer** module that captures long-range contextual features from voxel tokens and extracts fine-grained point features from point tokens. To be specific, the voxel tokens are obtained from non-empty voxels around reference points to cover a large attention range. In contrast, the point tokens are generated from neighboring points near reference points to retain fine-grained information. These two different tokens are adaptively fused by the cross-attention layer based on the similarity with content queries to complement each other.

Furthermore, we design a **Virtual Range Image** module to accelerate the neighbor querying process in the point-voxel Transformer. With the constructed range image, reference points can quickly find their neighbors based on range image coordinates. Unlike the native range image captured by LiDAR sensors, we can handle situations where multiple points overlap on the same pixel in the range image. Therefore, it can be used for complex scenarios, such as multiple sensors and multi-frame fusion.

Extensive experiments have been conducted on several detection benchmarks to verify the efficacy and efficiency of our approach. PVT-SSD achieves competitive results on KITTI [13], Waymo Open Dataset [48], and nuScenes [3].

2. Related Work

3D Object Detection from Point Clouds. Current 3D detectors can be mainly divided into voxel-, point-, and point-voxel-based methods. To process irregular 3D point clouds, voxel-based methods [7,16,53,56,61,74,75] project them onto regular voxels. VoxelNet [76] leverages PointNet [40] to generate a voxel-wise representation and applies standard 3D and 2D convolutions for object detection. PointPillars [21] simplifies the voxels to pillars. CenterPoint [70] estimates the centers of objects using a keypoint detector and removes the need for axis-aligned anchor boxes. Though voxel-based methods achieve good detection performance with promising efficiency, voxelization inevitably introduces quantization errors.

Point-based methods [45–47, 66] overcome this by directly operating on raw point clouds. VoteNet [39] detects 3D objects through Hough voting and clustering. 3DSSD [65] proposes a hybrid sampling strategy by utilizing both feature and geometry distance for better classification performance. Some approaches [6, 52, 72] use objectness scores rather than feature distance to improve the foreground points ratio after downsampling. It is generally time-consuming to repeatedly apply sampling and grouping operations on large-scale point clouds.

Point-voxel-based methods [35, 63, 69] take advantage of the efficiency of 3D sparse convolutions while preserving accurate point locations. PV-RCNN [43] and its variants [44] extract point-wise features from voxel abstraction networks to refine the proposals generated from the 3D voxel backbone. Pyramid R-CNN [32] collects points for each RoI in a pyramid manner.

3D Object Detection from Range Images. There are some prior works [2,24,50] that attempt to predict 3D boxes from raw representations captured by LiDAR sensors, i.e., 2D perspective range images. LaserNet [34] applies traditional 2D convolutions to range images to directly regress boxes. RangeDet [12] and PPC [5] introduce point-based convolution kernels to capture 3D geometric information from 2D range view representation. The representation of range images is compact and dense, and free of quantization errors, which inspires us to use it to speed up the ball query algorithm [41] widely used in point-based methods. The difference from earlier methods is that our constructed virtual range images can handle more complex situations, such as point clouds from multi-frame or multi-sensor.

Point Cloud Analysis by Transformer. Transformer [51] has demonstrated its great success in many computer vision tasks [4, 78]. Recent approaches [20, 25, 25, 28, 37, 38, 67, 73, 77] also explore it for point cloud analysis. Pointformer [37] proposes local and global attention modules to process 3D point clouds. Group-Free [29] eliminates hand-crafted grouping schemes by applying an attention module on all the points. 3DETR [36] develops an end-to-end Transformer-based detection model with minimal 3D specific inductive biases. VoTr [33] introduces a voxel-based Transformer that adopts both local and dilated attention to enlarge receptive fields of the model. SST [10] extends the shifted window [26] to 3D scenes and employs self-attention on non-empty voxels within the window. Object DGCNN [54] incorporates grid-based BEV features around queries through deformable attention [78]. VISTA [8] adaptively fuses global multi-view features via an attention module. Despite the effectiveness, they often fail to capture fine patterns of point clouds due to voxelization. CT3D [42] builds Transformer on top of a two-stage detector and operates attention on the points grouped by RoIs. EQ-PVRCNN [64] takes proposal grids as queries and generates RoI features from a voxel-based backbone.

3. Methodology

The architecture of our model is illustrated in Figure 1. Unlike previous Transformer-based indoor 3D detectors [29, 36], our target application is for outdoor scenes where point clouds are sparse and large-scale. Thus, we use sparse convolutions as the backbone for efficient feature encoding. Then the query initialization (in Sec. 3.1) obtains reference points from the non-empty voxels and content queries from the dense BEV feature map. After that, the point-voxel Transformer (in Sec. 3.2) samples voxels and points around reference points and utilizes their positions and features to capture contextual and geometric features via Transformer blocks. To quickly find neighboring points

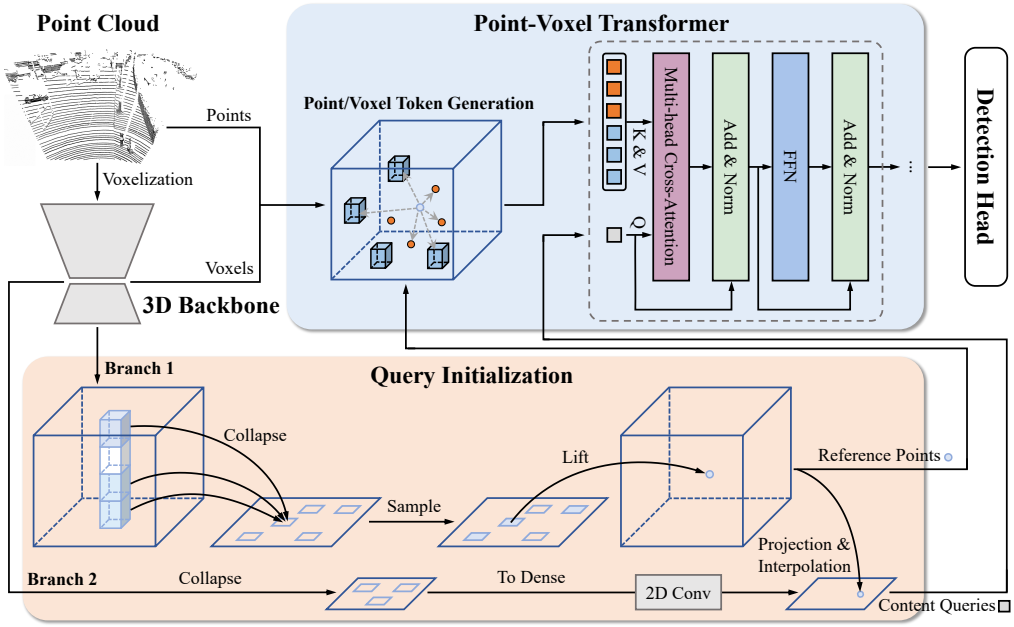

Figure 1. Architecture. The raw point clouds are voxelized to feed into the sparse convolutions. **Query Initialization** module obtains reference points by collapsing, sampling, and lifting non-empty voxels, and then attains the corresponding content queries from the dense BEV feature map. They are processed by the **Point-Voxel Transformer** to adaptively fuse voxel and point features. Finally, the detection head uses fused features to produce classification and regression.

near reference points, we propose the virtual range image (in Sec. 3.3) to speed up the process. Finally, we show the definition of the loss function (in Sec. 3.4).

3.1. Query Initialization

Recent methods [29, 36, 64] show that taking input-dependent queries (i.e., Q for the attention layer) benefits object detection. For example, 3DETR [36] for indoor 3D detection samples a set of reference points from inputs through FPS and associates each point with a content query. However, it is not friendly for outdoor scenes because sampling from large-scale points is time-consuming. To solve this problem, we propose a novel query initialization module that samples voxels instead of points to reduce the sampling time. Meanwhile, we make the voxels near objects more likely to be sampled, which can further boost the detection performance. Specifically, the module has two branches: branch 1 samples 2D non-empty voxels and lifts the 2D voxels to generate 3D reference points; branch 2 first constructs the dense feature map through several convolutional layers and then indexes features in terms of branch 1's coordinates as content queries.

Branch 1. For all input 3D voxels, we first collapse the voxels along the height dimension to further reduce the number of non-empty voxels. To keep efficiency, we adopt the max pooling for voxels locating the same horizontal location. After that, a representative set of voxels will be sampled. Ideally, we would like the sampled voxels to recall as many of the foreground objects as possible. Inspired by S-FPS [6], we predict whether voxels are in objects and sample some of them according to their probability and geometric distance. This allows voxels with better initial positions to be sampled and avoids sampled voxels being spatially too close. Concretely, the predicted probability is supervised by the foreground mask, i.e., if the voxel is inside a box, its label is set to 1; otherwise, it is 0. We then multiply the foreground probability by the original voxel distance as the new distance metric, and iteratively sample the most distant voxel. Finally, we use the center of objects as a guide to lift 2D voxels to 3D points. That is, for each sampled voxel, it predicts the offsets to the center of its corresponding 3D boxes, and the height of voxels is set to 0 by default. The predicted offsets are then added to the coordinates of voxels to acquire 3D reference points $\mathcal{P}_{\text{query}} \subset \mathbb{R}^3$.

Branch 2. As the generated reference points have deviated from the coordinates of the original voxels, direct use of the original voxel features as the corresponding content queries will result in a mismatch between instance features and positions. To overcome this, we use voxel features from the middle layer of the 3D backbone to align the new positions by projection and interpolation. Firstly, the 3D voxels are collapsed into the sparse BEV feature map, which is then converted to dense, i.e., empty voxels are filled with zeros. Subsequently, several lightweight convolutional layers are applied for feature extraction to avoid the generated reference points indexing invalid features from empty voxels. After that, we project reference points to 2D BEV coordinate system and index features on the BEV feature map

as content queries $\mathcal{F}_{\text{query}} \subset \mathbb{R}^d$. Since the projected coordinates may not be integers, we use bilinear interpolation to collect the feature vector.

3.2. Point-Voxel Transformer

The point-voxel Transformer takes four inputs: reference points, content queries, raw points, and voxels from the 3D backbone. The voxel tokens and point tokens are generated, which are then fed into several Transformer blocks to adaptively captures long-range contextual features and fine-grained geometric features.

Voxel Token Generation. To capture long-range contexts, some methods [8, 29, 37] perform attention on all points or sampled points, which are inefficient. The former has too many points (e.g., 180K for point clouds and 20K for non-empty voxels), while the latter relies on FPS for downsampling. Recent studies [36] show that local feature aggregation matters more than global aggregation. Motivated by that, to reduce computational and memory overheads while keeping large receptive fields, we randomly sample l voxels from the middle layer of the 3D backbone within a large radius r_v (i.e., 8m) of reference points as the voxel tokens, which consists of voxel coordinates $\mathcal{P}_{\text{voxel}}$ and voxel features $\mathcal{F}_{\text{voxel}}$.

Point Token Generation. Due to the quantization artifacts produced during voxelization, a great hurdle remains for the voxel tokens in producing accurate 3D boxes. Therefore, we generate the point tokens, which have smaller receptive fields than the voxel tokens but can provide fine-grained point features. Specifically, we apply the ball query (introduced in Sec. 3.3) to sample l points within the radius r_p (i.e., 3.2m) of reference points. However, the sampled points $\mathcal{P}_{\text{point}}$ only contain xyz position information, lacking local geometric and contextual information. As a result, the attention map fails to capture the high-level correlation between the query and key in the Transformer block. Inspired by [16, 27], we interpolate the features of 3D voxels near the points to obtain point features $\mathcal{F}_{\text{point}}$:

$$\left\{ f_i = \frac{\sum_{j=1}^{k} w_j^i \cdot \bar{f}_j^i}{\sum_{j=1}^{k} w_j^i} \;\middle|\; w_j^i = \frac{1}{\left\| \bar{p}_j^i - p_i \right\|_2} \right\}, \quad (1)$$

where $p_i \in \mathcal{P}_{\text{point}}$ and $f_i \in \mathcal{F}_{\text{point}}$ are the coordinate and feature of the i-th sampled point, respectively, $\bar{p}_j^i$ and $\bar{f}_j^i$ are the center and feature of the j-th voxel near p_i, respectively, and k is the number of neighbors. We use k-nearest neighbors (KNN) to acquire $\bar{p}_j^i$ for each p_i. However, the original KNN implemented by PyTorch costs $O(mln)$ to find neighbors, as shown in Figure 2(a). Inspired by [7], we propose the voxel-based KNN that only searches nearby voxels instead of all of them to reduce the complexity to $O(mlv^3)$. To further diminish the computational costs, we design the conquer-fetch operation (in (ii) of Figure 2(b)). We empirically observe that some redundant points are sampled for

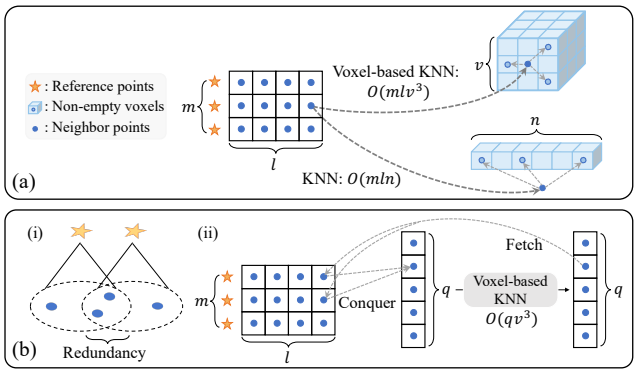

Figure 2. Illustration of the voxel-based KNN (a) and the conquer-fetch operation (b).

different reference points (in (i) of Figure 2(b)), but their voxel neighbors are the same. Therefore, the conquer operation is first applied to remove redundancy, followed by the voxel-based KNN to find voxel neighbors. The fetch operation is then applied to obtain voxel neighbors of all points. The time complexity is further decreased to $O(qv^3)$.

Transformer Block. Given matrices of the voxel tokens $\mathcal{F}_{\text{voxel}}$ and $\mathcal{P}_{\text{voxel}}$, and the point tokens $\mathcal{F}_{\text{point}}$ and $\mathcal{P}_{\text{point}}$, we first concatenate voxel tokens and point tokens to construct $\mathcal{F}_s = [\mathcal{F}_{\text{voxel}}, \mathcal{F}_{\text{point}}]$ and $\mathcal{P}_s = [\mathcal{P}_{\text{voxel}}, \mathcal{P}_{\text{point}}]$. Combined with the content queries $\mathcal{F}_{\text{query}}$ and the reference points $\mathcal{P}_{\text{query}}$, they are then fed into a Transformer block:

$$\begin{aligned} X &= \text{Attention}(\mathcal{F}_s, \mathcal{F}_{\text{query}}, \mathcal{P}_s, \mathcal{P}_{\text{query}}) + \mathcal{F}_{\text{query}}, \\ Y &= \text{FFN}(X) + X, \end{aligned} \quad (2)$$

where Attention is a multi-head cross-attention layer with contextual relative positional encoding [55, 64], and FFN is a feed-forward network. We use LayerNorm [1] to normalize features after each Attention and FFN module.

3.3. Virtual Range Image

As mentioned in point token generation (in Sec. 3.2), we introduce a novel ball query to quickly find neighbors of reference points. As illustrated in Figure 3, our ball query is based on the virtual range image that is constructed from the point clouds of multi-sensor and multi-frame.

Setup. Let $\mathcal{R}^i_{\text{lidar}\to\text{car}} \in \mathbb{R}^{4\times4}$ be a homogeneous transformation matrix that transforms points $\mathcal{P}^i_{\text{lidar}} \subset \mathbb{R}^3$ of each LiDAR sensor $\mathcal{S}_i$ from the sensor coordinate system to the car coordinate system. As a common practice, we use all points $\mathcal{P}_{\text{car}}$ from five LiDAR sensors on the Waymo dataset:

$$\mathcal{P}_{\text{car}} = \left\{ \mathcal{P}^i_{\text{lidar}} \cdot \mathcal{R}^i_{\text{lidar}\to\text{car}} \mid i \in \{0, \dots, 4\} \right\}. \quad (3)$$

Inverse Augmentation. In 3D object detection, augmentation (e.g., copy-n-paste [60], global rotation, and random flip) plays a vital role in reducing model overfitting. For copy-n-paste, the object points $\mathcal{P}_{\text{gt}}$ from other frames are pasted on the current frame with their original 3D positions. We combine $\mathcal{P}_{\text{gt}}$ and $\mathcal{P}_{\text{car}}$ to get a new point set

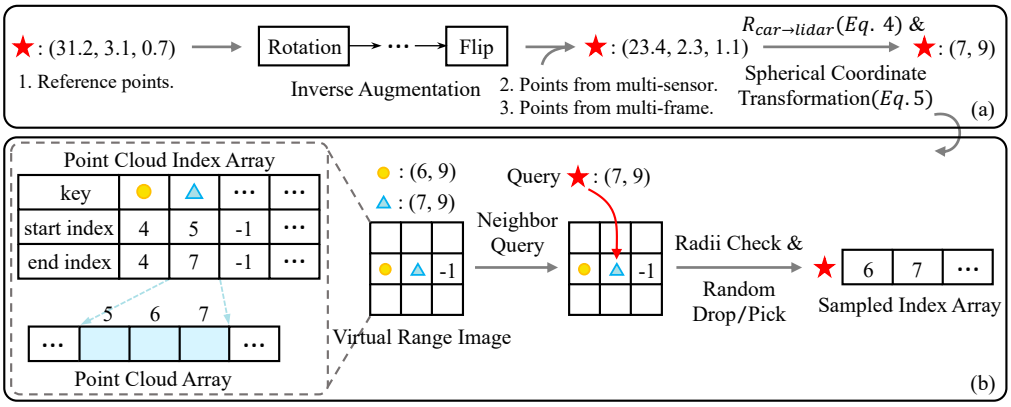

Figure 3. Illustration of the proposed ball query, which consists of two parts: (a) Inverse augmentation and coordinate transformation; (b) We construct the virtual range image where the pixel value is represented as Point Cloud Index Array. The index array records the overlapped 3D points on the range image. Then we use Neighbor Query to find the neighbors of reference points, and apply Radii Check and Random Drop/Pick to sample 3D points.

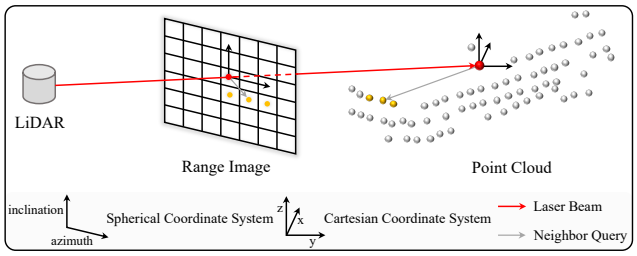

Figure 4. Illustration of the Cartesian and the Spherical coordinate system of the LiDAR sensor.

$\mathcal{P}$, which will be used to construct the virtual range image. For other geometry-related data augmentation, we save augmentation parameters (e.g., the rotation angle for global rotation). Since the reference points $\mathcal{P}_{\text{query}}$ (we use $\mathcal{Q}$ to simplify the notation) are generated after augmentation, we reverse all those data augmentations on $\mathcal{Q}$ to get the original coordinate (in Figure 3(a)), which is similar to [22, 71].

Coordinate Transformation. Next, we transform $\mathcal{P}$ and $\mathcal{Q}$ from the reference frame of the car back to the top LiDAR sensor (i.e., $\mathcal{S}_0$):

$$\mathcal{P}' = \mathcal{P} \cdot \mathcal{R}^0_{\text{car}\to\text{lidar}}, \quad \mathcal{Q}' = \mathcal{Q} \cdot \mathcal{R}^0_{\text{car}\to\text{lidar}}, \tag{4}$$

where $\mathcal{P}'$ and $\mathcal{Q}'$ are the coordinates in the LiDAR Cartesian coordinate system. For each point $\boldsymbol{p} = (x, y, z) \in \mathcal{P}' \cup \mathcal{Q}'$, it is uniquely transformed to the LiDAR Spherical coordinate system by the following equations:

$$\begin{aligned} &\theta = \text{atan2}(y, x), \quad \phi = \text{atan2}(z, \sqrt{x^2 + y^2}), \\ &r = \sqrt{x^2 + y^2 + z^2}, \end{aligned} \tag{5}$$

where θ, ϕ, and r indicate the laser's azimuth angle, inclination angle, and range, respectively.

Image Construction. Suppose $\bar{\mathcal{P}}$ and $\bar{\mathcal{Q}}$ are transformed coordinates of $\mathcal{P}'$ and $\mathcal{Q}'$, respectively. The native range image uses θ as the column index, ϕ as the row index, and r as the pixel value, which has a one-to-one correspondence with the point cloud captured by one LiDAR sensor, as shown in Figure 4. However, as $\bar{\mathcal{P}}$ is from multiple frames as well as multiple LiDAR sensors, various points may overlap on the same pixel. To overcome this issue, each pixel of our constructed virtual range image is an array of indices pointing to overlapped 3D points. For convenience, we organize overlapped points adjacently and record their start and end position indices, as shown by Point Cloud Index Array and Point Cloud Array in Figure 3(b).

Random Neighbor Query. With the constructed image, each reference point in $\bar{\mathcal{Q}}$ can quickly find its neighbors because θ and ϕ determine the pixel position in the virtual range image, as shown by the Neighbor Query in Figure 3(b). To prevent the distance between adjacent pixels from being large in 3D space, we will check whether the neighbors are within the radius of the reference point (i.e., Radii Check). During the process, we will apply a random selection algorithm[1] to retain at most k neighbors (i.e., Random Drop/Pick). The overall operation can be efficiently executed in parallel on the GPU. In this way, we reduce the theoretical time complexity of the original ball query [41] from $\mathcal{O}(mn)$ to $\mathcal{O}(ms^2)$, where m is the number of reference points, n is the number of point clouds, and s is the kernel size to visit neighbors in the virtual range image. Notably, s^2 is much smaller than n.

3.4. Loss Function

The overall loss consists of the segmentation loss, offset loss, classification loss, and regression loss.

Segmentation Loss & Offset Loss. We regard non-empty voxels inside any ground-truth bounding boxes as foreground voxels. Since voxels are two-dimensional, the height of the ground-truth box is not considered when as-

[1] https://github.com/LeviViana/torch_sampling

signing labels. The segmentation loss is computed by:

$$\mathcal{L}_{\text{seg}} = \frac{1}{N^v} \cdot \sum_{i=1}^{N^v} \text{CE}\left(s_i, \hat{s}_i\right), \tag{6}$$

where N^v is the number of non-empty voxels, and CE is the cross entropy loss function. The offset loss is calculated by:

$$\mathcal{L}_{\text{offset}} = \frac{1}{N_+^v} \cdot \sum_{i=1}^{N_+^v} \text{L1}\left(o_i, \hat{o}_i\right), \tag{7}$$

where N_+^v is the number of foreground voxels, and L1 is the smooth-l_1 loss function.

Classification Loss & Regression Loss. We adopt the same target assignment strategy and prediction head following [6, 65, 72]. Specifically, for each reference point, we consider the point inside an annotated bounding box as the foreground point and then compute the centerness [65] as its label. The classification loss is:

$$\mathcal{L}_{\text{cls}} = \frac{1}{N^q} \cdot \sum_{i=1}^{N^q} \text{CE}\left(c_i, \hat{c}_i\right), \tag{8}$$

where N^q is the number of reference points. For the regression loss $\mathcal{L}_{\text{reg}}$, we decouple it to center regression loss, box size estimation loss, and heading angle estimation loss. We refer readers to [65] for more details.

4. Experiments

4.1. Datasets

Waymo Open Dataset [48] is a large-scale autonomous driving dataset consisting of 798 scenes for training and 202 scenes for validation. The evaluation protocol consists of the average precision (AP) and average precision weighted by heading (APH). Also, it includes two difficulty levels: LEVEL_1 denotes objects containing more than 5 points, and LEVEL_2 denotes objects containing at least 1 point. To save training time, we use a subset of the training splits by sampling every 10 frames for ablation studies.

KITTI [13] contains 7481 training samples and 7518 testing samples in autonomous driving scenes. As a common practice, the training data are divided into a *train* set with 3712 samples and a *val* set with 3769 samples.

nuScenes [3] is a challenging dataset for autonomous driving with 380K LiDAR sweeps from 1000 scenes. The evaluation metrics used in nuScenes dataset incorporate the commonly used mean average precision (mAP) and a novel nuScenes detection score (NDS).

4.2. Implementation Details

Our implementation is based on the codebase of OpenPCDet[2]. For the Waymo dataset, the detection ranges are set as $(-75.2, 75.2)$, $(-75.2, 75.2)$, and $(-2, 4)$, and the voxel size is $(0.1m, 0.1m, 0.15m)$. We adopt a similar 3D sparse backbone as [68], but the last two upsampling layers are removed to keep efficiency. For the query initialization, we apply 4 traditional 2D convolutional layers with dimensions 64 on the BEV feature map; we sample 512 voxels to generate reference points. For the point-voxel Transformer, we sample 128 neighbors for each reference point and apply

[2] https://github.com/open-mmlab/OpenPCDet

Table 1. Performance comparison on the Waymo validation set for vehicle class. 3f: taking 3 frames as input. The results achieved by our PVT-SSD are shown in bold, while the top-performed results are shown in underline.

Methods	LEVEL_1 3D AP/APH	LEVEL_2 3D AP/APH
Two-stage:		
RSN [50]	75.10/74.60	66.00/65.50
Pyramid RCNN [32]	76.30/75.68	67.23/66.68
SST_TS [10]	76.22/75.79	68.04/67.64
LiDAR R-CNN [23]	76.00/75.50	68.30/67.90
Part-A2-Net [46]	77.05/76.51	68.47/67.97
CenterPoint-Voxel [70]	76.70/76.20	68.80/68.30
PV-RCNN [43]	77.51/76.89	68.98/68.41
CT3D [42]	76.30/-	69.04/-
PDV [18]	76.85/76.33	69.30/68.81
BtcDet [57]	78.58/78.06	70.10/69.61
PV-RCNN++ [44]	79.25/78.78	70.61/70.18
One-stage:		
IA-SSD [72]	70.53/69.67	61.55/60.80
PointPillars [21]	71.56/70.99	63.05/62.54
SECOND [60]	72.27/71.69	63.85/63.33
RangeDet [12]	72.90/72.30	64.00/63.60
CenterPoint-Pillar [70]	73.37/72.86	65.09/64.62
SST [10]	74.22/73.77	65.47/65.07
VoxSeT [15]	74.50/74.03	65.99/65.56
CenterPoint-Voxel [70]	74.78/74.24	66.66/66.17
Point2Seq [59]	77.52/77.03	68.80/68.36
MsSVT [9]	77.83/77.32	69.53/69.06
SWFormer [49]	77.80/77.30	69.20/68.80
SWFormer_3f [49]	79.40/78.90	71.10/70.60
PVT-SSD (Ours)	**79.16/78.72**	**70.23/69.83**
PVT-SSD_3f (Ours)	**80.59/80.16**	**71.86/71.47**

Table 2. Performance comparison on the Waymo validation set for pedestrian and cyclist classes.

Methods	Pedestrian	Cyclist
Two-stage:		
LiDAR R-CNN [23]	63.10/51.70	66.10/64.40
PV-RCNN [43]	66.04/57.61	65.39/63.98
PDV [18]	65.85/58.28	66.49/65.36
Part-A2-Net [46]	66.18/58.62	66.13/64.93
RSN [50]	68.30/63.70	-/-
PV-RCNN++ [44]	73.17/68.00	71.21/70.19
One-stage:		
SST [10]	70.02/61.67	-/-
CenterPoint-Voxel [70]	68.42/62.67	69.69/68.59
RangeDet [12]	67.60/63.90	63.30/62.10
VoxSeT [15]	72.45/65.39	68.95/67.73
MsSVT [9]	73.00/66.65	72.37/71.24
SWFormer [49]	72.50/64.90	-/-
SWFormer_3f [49]	74.80/71.10	-/-
PVT-SSD (Ours)	**72.56/67.02**	**73.94/72.96**
PVT-SSD_3f (Ours)	**75.11/72.12**	**74.80/73.97**

Table 3. Performance comparison on the Waymo leaderboard.

Methods	mAP/mAPH	Vehicle	Pedestrian	Cyclist
PV-RCNN [43]	71.25/68.75	72.81/72.39	71.81/66.05	69.13/67.80
PV-RCNN++ [44]	72.42/70.20	73.86/73.47	74.12/69.00	69.28/68.15
Graph-RCNN [63]	73.81/71.59	76.04/75.64	75.59/70.45	69.79/68.67
GD-MAE [62]	74.71/72.29	75.83/75.46	77.10/71.28	71.21/70.15
FSD [11]	74.39/72.35	74.40/74.06	75.93/71.26	72.85/71.75
CenterPoint_2f [70]	73.38/71.93	73.42/72.99	74.56/71.52	72.17/71.28
SST_TS_3f [10]	74.41/72.81	73.08/72.74	76.93/73.51	73.22/72.17
SWFormer_3f [49]	-/-	75.02/74.65	75.87/72.07	-/-
PVT-SSD (Ours)	**72.73/70.44**	**72.96/72.62**	**73.15/67.72**	**72.08/70.97**
PVT-SSD_3f (Ours)	**74.69/73.28**	**75.24/74.89**	**75.62/72.60**	**73.21/72.35**

one Transformer block; the r_v and r_p are set to 8.0 and 3.2, respectively; for each Transformer block, the input dimension, the hidden dimension, the number of head, and the dropout are set to 128, 512, 4, and 0.1, respectively; for the point token generation, each point interpolates from voxel features of the 8 nearest neighbors. We train the model for 30 epochs with the AdamW [31] optimizer using the one-cycle policy, with a max learning rate of $3e^{-3}$.

4.3. Comparison with State-of-the-Art Methods

We compare PVT-SSD on the Waymo validation set with previous methods in Table 1 and Table 2. We divide current methods into the branches of one-stage and two-stage detectors for comparisons. Table 1 shows the results on vehicles, where PVT-SSD surpasses previous one-stage methods with a single frame LiDAR input. The performance is also comparable with state-of-the-art two-stage methods. Table 2 shows results on pedestrians and cyclists. PVT-SSD outperforms the strong single-stage detector, i.e., CenterPoint-Voxel, by 4.35 APH and 4.37 APH for the pedestrian and cyclist, respectively. The performance can be further improved by 5.1 APH and 1.01 APH, respectively, when taking 3 frames as input. Table 3 illustrates the performance on the Waymo test set. We achieve competitive results compared with previous methods. Table 4 illustrates the performance comparisons on the official KITTI test server. It shows that PVT-SSD has the best car detection performance among all single-stage detectors at both easy and moderate levels. Compared with previous Transformer-based detectors, PVT-SSD is 1.4× and 4.4× faster in inference speed than CT3D and VoTr-TSD, respectively, and achieves better moderate AP. It also requires less number of parameters, as shown in Table 5. In Table 6, we report the results on nuScenes validation set. Our method obtains much higher NDS and mAP compared with SASA [6].

4.4. Ablation Study

In this section, we conduct a series of ablation experiments to comprehend the roles of different components.

Query Initialization. Lifting plays an essential role in query initialization. It can not only make 2D voxels into 3D points, but also can bring reference points closer to the center of the object, which can make it easier for subsequent regressions to recall their corresponding 3D boxes. As shown in the first and second rows of Table 7, in combination with feature alignment, it brings 0.61 APH gains. In Table 8, we compare performances when using different sampling strategies. Compared with FPS and F-FPS [65], S-FPS yields 11.25 APH and 0.48 APH benefits, respectively. It considers the probability of whether voxels are in objects when sampling, which allows it to sample voxels with better initial positions to generate reference points. S-FPS is also 0.81 APH higher than Top-K because it avoids sampling voxels too closely in space, thus recalling foreground objects as much as possible. The fourth and sixth rows of Table 7 show that feature alignment leads to a 0.87 APH improvement because the semantic similarity with voxel tokens and point tokens can be calculated more correctly.

Point-Voxel Transformer. Table 7 shows that the voxel tokens and the point tokens provide improvements of 0.45 and 1.58 APH, respectively. We find that the improvement brought by the point tokens is larger than that of the voxel tokens, which is intuitive since fine-grained point features are essential for accurate box regression. Table 9 ablates the influence of r_v and r_p, which will affect the receptive field. We observe that large receptive fields improve detection performance, but it is not further improved when con-

Table 4. Performance comparison on the KITTI testing sever for the car class. †: the latency is measured by us based on the same environment (i.e., RTX 3080Ti GPU) with 1 batch size.

Methods	3D AP			BEV AP			Latency
	Easy	Moderate	Hard	Easy	Moderate	Hard	(ms)
Two-stage:							
PointRCNN [45]	86.96	75.64	70.70	92.13	87.39	82.72	100
STD [66]	87.95	79.71	75.09	94.74	89.19	86.42	80
PV-RCNN [43]	90.25	81.43	76.82	94.98	90.65	86.14	98†
M3DETR [14]	90.28	81.73	76.96	94.41	90.37	85.98	-
CT3D [42]	87.83	81.77	77.16	92.36	88.83	84.07	70†
PDV [18]	90.43	81.86	77.36	94.56	90.48	86.23	135
PV-RCNN++ [44]	90.14	81.88	77.15	92.66	88.74	85.97	60
EQ-PVRCNN [64]	90.13	82.01	77.53	94.55	89.09	86.42	200
Pyramid-PV [32]	88.39	82.08	77.49	92.19	88.84	86.21	127
VoTr-TSD [33]	89.90	82.09	79.14	94.03	90.34	86.14	216†
SPG [58]	90.50	82.13	78.90	94.33	88.70	85.98	156
One-stage:							
SECOND [60]	83.34	72.55	65.82	89.39	83.77	78.59	50
PointPillars [21]	82.58	74.31	68.99	90.07	86.56	82.81	24
TANet [30]	84.39	75.94	68.82	91.58	86.54	81.19	35
Point-GNN [47]	88.33	79.47	72.29	93.11	89.17	83.90	643
3DSSD [65]	88.36	79.57	74.55	92.66	89.02	85.86	38
SA-SSD [16]	88.75	79.79	74.16	95.03	91.03	85.96	40
IA-SSD [72]	88.87	80.32	75.10	93.14	89.48	84.42	44†
SVGA-Net [17]	87.33	80.47	75.91	92.07	89.88	85.59	62
PVGNet [35]	89.94	81.81	77.09	94.36	91.26	86.63	-
SASA [6]	88.76	82.16	77.16	92.87	89.51	86.35	63†
PVT-SSD (Ours)	**90.65**	**82.29**	**76.85**	**95.23**	**91.63**	**86.43**	**49†**

Table 5. Comparisons of the number of parameters.

Methods	CT3D [42]	VoTr-TSD [33]	M3DETR [14]	**PVT-SSD**
# Param.	30M	49M	76M	**16M**

Table 6. Performance comparison on the nuScenes validation set. †: we follow [6, 65] to predict all classes in a single head.

Methods	NDS	mAP	Car	Truck	Bus	Trailer	C. V.	Ped.	Motor	Bicycle	T. C.	Barrier
SECOND [60]	-	27.1	75.5	21.9	29.0	13.0	0.4	59.9	16.9	0	22.5	32.2
PointPillars [21]	44.9	29.5	70.5	25.0	34.4	20.0	4.5	59.9	16.7	1.6	29.6	33.2
3DSSD [65]	56.4	42.7	81.2	47.2	61.4	30.5	12.6	70.2	36.0	8.6	31.1	47.9
SASA [6]	61.0	45.0	76.8	45.0	66.2	36.5	16.1	69.1	39.6	16.9	29.9	53.6
PVT-SSD† (Ours)	**65.0**	**53.6**	**79.4**	**43.8**	**62.1**	**34.2**	**21.7**	**79.8**	**53.4**	**38.2**	**56.6**	**67.1**

Table 7. Ablations on the Waymo validation set for vehicle class.

Lift.	Alig.	Voxel Tok.	Point Tok.	3D AP/APH
				62.82/62.37
✓	✓			63.40/62.98
✓	✓	✓		63.85/63.43
✓		✓	✓	64.57/64.14
✓	✓		✓	65.12/64.70
✓	✓	✓	✓	**65.45/65.01**

Table 8. Ablations of sampling strategies in query initialization.

Methods	S-FPS	F-FPS	FPS	Top-K
APH	**65.01**	64.53	53.76	64.20

Table 9. Ablations of radii r_v and r_p in point-voxel Transformer.

r_v	6.4	8.0	9.6
APH	64.79	**65.01**	64.94

r_p	1.6	3.2	4.8
APH	64.21	**65.01**	64.97

tinuing to increase the receptive field. It may be due to the fact that more irrelevant noise points are sampled while relevant points benefiting detection are randomly discarded. Table 10 shows the importance of contextual relative position encoding, which has a huge impact on the learned representation. When absolute position encoding [36] and bias-mode relative position encoding [55] are used, the APH drops by 1.25 and 0.73, respectively.

Virtual Range Image. Table 11 illustrates the comparison between random and sequential sampling used in the range-view-based ball query. Sequential sampling means that we traverse the kernel in order from top-left to bottom-right and sample points within the radius until the number of points is satisfied. We observe an APH drop of 0.9 when using sequential sampling since it may result in an uneven distribution of sampled points, i.e., points in the top-left part are more likely to be sampled. Figure 5 compares the latency of our range-view-based ball with that of the original ball query, where the kernel size is set to 16 to find 32 neighbors within a radius of 0.8. It shows that the range view-based ball query performs well as the point cloud increases and achieves a speedup of 29.7× over the original ball query when it is applied on 200K point clouds.

Table 10. Ablations of positional encoding in Transformer blocks.

Methods	None	Relative Context	Relative Bias	Absolute
APH	63.29	**65.01**	64.28	63.76

Table 11. Ablation study of the random and the sequential sampling in RV-based ball query.

Methods	Random	Sequential
APH	**65.01**	64.11

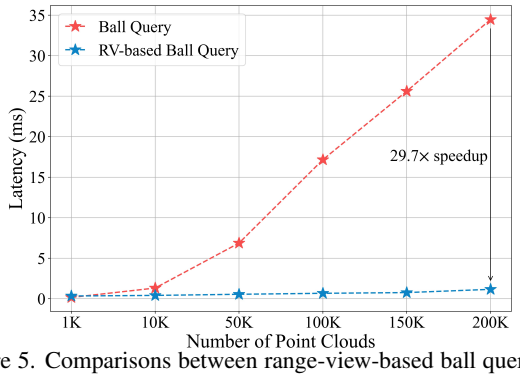

Figure 5. Comparisons between range-view-based ball query and original ball query.

Visualization. Figure 6 shows the attention weights of the sampled tokens for each reference point. The model can capture object-centric features and long-range features.

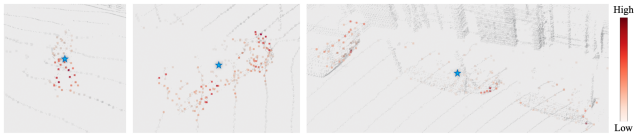

Figure 6. Visualization of the attention map.

5. Conclusion

PVT-SSD leverages the benefits from the voxel and point representations. We propose the Query Initialization module to generate reference points and content queries to associate these two different representations. Then, the Point-Voxel Transformer module is introduced to capture long-range contextual features from voxels and fine-grained geometric features from points. To accelerate the neighbor querying process, we design a Virtual Range Image module. The constructed range image is a generalized version of the native range image captured by LiDAR sensors and thus can be used for more scenarios. Experiments on several autonomous driving benchmarks demonstrate the efficacy.

Acknowledgement This work was supported in part by The National Nature Science Foundation of China (Grant Nos: U1909203, 62273303), in part by Innovation Capability Support Program of Shaanxi (Program No. 2021TD-05), in part by the Key R&D Program of Zhejiang Province, China (2023C01135), in part by S&T Plan of Zhejiang Province (No. 202218), in part by the National Key R&D Program of China (No. 2022ZD0160100), and in part by Shanghai Committee of Science and Technology (Grant No. 21DZ1100100).

References

[1] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016. 4

[2] Alex Bewley, Pei Sun, Thomas Mensink, Drago Anguelov, and Cristian Sminchisescu. Range conditioned dilated convolutions for scale invariant 3d object detection. In *Conference on Robot Learning*, 2020. 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 Conference on Computer Vision and Pattern Recognition*, 2020. 2, 6

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision*, 2020. 2

[5] Yuning Chai, Pei Sun, Jiquan Ngiam, Weiyue Wang, Benjamin Caine, Vijay Vasudevan, Xiao Zhang, and Dragomir Anguelov. To the point: Efficient 3d object detection in the range image with graph convolution kernels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2

[6] Chen Chen, Zhe Chen, Jing Zhang, and Dacheng Tao. SASA: semantics-augmented set abstraction for point-based 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022. 2, 3, 6, 7, 8

[7] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021. 2, 4

[8] Shengheng Deng, Zhihao Liang, Lin Sun, and Kui Jia. Vista: Boosting 3d object detection via dual cross-view spatial attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2, 4

[9] Shaocong Dong, Lihe Ding, Haiyang Wang, Tingfa Xu, Xinli Xu, Jie Wang, Ziyang Bian, Ying Wang, and Jianan Li. Mssvt: Mixed-scale sparse voxel transformer for 3d object detection on point clouds. In *Advances in Neural Information Processing Systems*, 2022. 6

[10] Lue Fan, Ziqi Pang, Tianyuan Zhang, Yu-Xiong Wang, Hang Zhao, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Embracing single stride 3d object detector with sparse transformer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2, 6, 7

[11] Lue Fan, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Fully sparse 3d object detection. In *Advances in Neural Information Processing Systems*, 2022. 7

[12] Lue Fan, Xuan Xiong, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Rangedet: In defense of range view for lidar-based 3d object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 2, 6

[13] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 2, 6

[14] Tianrui Guan, Jun Wang, Shiyi Lan, Rohan Chandra, Zuxuan Wu, Larry Davis, and Dinesh Manocha. M3DETR: multi-representation, multi-scale, mutual-relation 3d object detection with transformers. In *Winter Conference on Applications of Computer Vision*, 2022. 7

[15] Chenhang He, Ruihuang Li, Shuai Li, and Lei Zhang. Voxel set transformer: A set-to-set approach to 3d object detection from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 6

[16] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3d object detection from point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2, 4, 7

[17] Qingdong He, Zhengning Wang, Hao Zeng, Yi Zeng, and Yijun Liu. Svga-net: Sparse voxel-graph attention network for 3d object detection from point clouds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022. 7

[18] Jordan S. K. Hu, Tianshu Kuai, and Steven L. Waslander. Point density-aware voxels for lidar 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 6, 7

[19] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 1

[20] Xin Lai, Jianhui Liu, Li Jiang, Liwei Wang, Hengshuang Zhao, Shu Liu, Xiaojuan Qi, and Jiaya Jia. Stratified transformer for 3d point cloud segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2

[21] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2, 6, 7, 8

[22] Yingwei Li, Adams Wei Yu, Tianjian Meng, Benjamin Caine, Jiquan Ngiam, Daiyi Peng, Junyang Shen, Bo Wu, Yifeng Lu, Denny Zhou, Quoc V. Le, Alan L. Yuille, and Mingxing Tan. Deepfusion: Lidar-camera deep fusion for multi-modal 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 5

[23] Zhichao Li, Feng Wang, and Naiyan Wang. Lidar r-cnn: An efficient and universal 3d object detector. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 6

[24] Zhidong Liang, Zehan Zhang, Ming Zhang, Xian Zhao, and Shiliang Pu. Rangeioudet: Range image based real-time 3d object detector optimized by intersection over union. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2

[25] Jiaheng Liu, Tong He, Honghui Yang, Rui Su, Jiayi Tian, Junran Wu, Hongcheng Guo, Ke Xu, and Wanli Ouyang. 3d-queryis: A query-based framework for 3d instance segmentation. *CoRR*, abs/2211.09375, 2022. 2

[26] 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 International Conference on Computer Vision*, 2021. 2

[27] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel CNN for efficient 3d deep learning. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems*, 2019. 4

[28] Zili Liu, Guodong Xu, Honghui Yang, Minghao Chen, Kuoliang Wu, Zheng Yang, Haifeng Liu, and Deng Cai. Suppress-and-refine framework for end-to-end 3d object detection. *CoRR*, abs/2103.10042, 2021. 2

[29] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-free 3d object detection via transformers. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 1, 2, 3, 4

[30] Zhe Liu, Xin Zhao, Tengteng Huang, Ruolan Hu, Yu Zhou, and Xiang Bai. Tanet: Robust 3d object detection from point clouds with triple attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020. 7

[31] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 7

[32] Jiageng Mao, Minzhe Niu, Haoyue Bai, Xiaodan Liang, Hang Xu, and Chunjing Xu. Pyramid r-cnn: Towards better performance and adaptability for 3d object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 2, 6, 7

[33] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. Voxel transformer for 3d object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 1, 2, 7

[34] Gregory P. Meyer, Ankit Laddha, Eric Kee, Carlos Vallespi-Gonzalez, and Carl K. Wellington. Lasernet: An efficient probabilistic 3d object detector for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2

[35] Zhenwei Miao, Jikai Chen, Hongyu Pan, Ruiwen Zhang, Kaixuan Liu, Peihan Hao, Jun Zhu, Yang Wang, and Xin Zhan. Pvgnet: A bottom-up one-stage 3d object detector with integrated multi-level features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2, 7

[36] Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3d object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 1, 2, 3, 4, 8

[37] Xuran Pan, Zhuofan Xia, Shiji Song, Li Erran Li, and Gao Huang. 3d object detection with pointformer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 1, 2, 4

[38] Chunghyun Park, Yoonwoo Jeong, Minsu Cho, and Jaesik Park. Fast point transformer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2

[39] Charles R. Qi, Or Litany, Kaiming He, and Leonidas J. Guibas. Deep hough voting for 3d object detection in point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 2

[40] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2

[41] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, 2017. 1, 2, 5

[42] Hualian Sheng, Sijia Cai, Yuan Liu, Bing Deng, Jianqiang Huang, Xian-Sheng Hua, and Min-Jian Zhao. Improving 3d object detection with channel-wise transformer. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 2, 6, 7

[43] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2, 6, 7

[44] Shaoshuai Shi, Li Jiang, Jiajun Deng, Zhe Wang, Chaoxu Guo, Jianping Shi, Xiaogang Wang, and Hongsheng Li. PV-RCNN++: point-voxel feature set abstraction with local vector representation for 3d object detection. *CoRR*, abs/2102.00463, 2021. 2, 6, 7

[45] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2, 7

[46] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 2, 6

[47] Weijing Shi and Raj Rajkumar. Point-gnn: Graph neural network for 3d object detection in a point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2, 7

[48] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2, 6

[49] Pei Sun, Mingxing Tan, Weiyue Wang, Chenxi Liu, Fei Xia, Zhaoqi Leng, and Dragomir Anguelov. Swformer: Sparse window transformer for 3d object detection in point clouds. In *Proceedings of the European Conference on Computer Vision*, 2022. 6, 7

[50] Pei Sun, Weiyue Wang, Yuning Chai, Gamaleldin Elsayed, Alex Bewley, Xiao Zhang, Cristian Sminchisescu, and

Dragomir Anguelov. RSN: range sparse net for efficient, accurate lidar 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2, 6

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017. 1, 2

[52] Haiyang Wang, Shaoshuai Shi, Ze Yang, Rongyao Fang, Qi Qian, Hongsheng Li, Bernt Schiele, and Liwei Wang. Rbgnet: Ray-based grouping for 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2

[53] Yue Wang, Alireza Fathi, Abhijit Kundu, David A. Ross, Caroline Pantofaru, Thomas A. Funkhouser, and Justin Solomon. Pillar-based object detection for autonomous driving. In *Proceedings of the European Conference on Computer Vision*, 2020. 2

[54] Yue Wang and Justin Solomon. Object DGCNN: 3d object detection using dynamic graphs. In *Advances in Neural Information Processing Systems*, 2021. 2

[55] Kan Wu, Houwen Peng, Minghao Chen, Jianlong Fu, and Hongyang Chao. Rethinking and improving relative position encoding for vision transformer. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 4, 8

[56] Xiaopei Wu, Liang Peng, Honghui Yang, Liang Xie, Chenxi Huang, Chengqi Deng, Haifeng Liu, and Deng Cai. Sparse fuse dense: Towards high quality 3d detection with depth completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2

[57] Qiangeng Xu, Yiqi Zhong, and Ulrich Neumann. Behind the curtain: Learning occluded shapes for 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022. 6

[58] Qiangeng Xu, Yin Zhou, Weiyue Wang, Charles R. Qi, and Dragomir Anguelov. SPG: unsupervised domain adaptation for 3d object detection via semantic point generation. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 7

[59] Yujing Xue, Jiageng Mao, Minzhe Niu, Hang Xu, Michael Bi Mi, Wei Zhang, Xiaogang Wang, and Xinchao Wang. Point2seq: Detecting 3d objects as sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 6

[60] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10), 2018. 4, 6, 7, 8

[61] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2

[62] Honghui Yang, Tong He, Jiaheng Liu, Hua Chen, Boxi Wu, Binbin Lin, Xiaofei He, and Wanli Ouyang. GD-MAE: generative decoder for MAE pre-training on lidar point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2023. 7

[63] Honghui Yang, Zili Liu, Xiaopei Wu, Wenxiao Wang, Wei Qian, Xiaofei He, and Deng Cai. Graph R-CNN: towards accurate 3d object detection with semantic-decorated local graph. In *Proceedings of the European Conference on Computer Vision*, 2022. 2, 7

[64] Zetong Yang, Li Jiang, Yanan Sun, Bernt Schiele, and Jiaya Jia. A unified query-based paradigm for point cloud understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2, 3, 4, 7

[65] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2, 6, 7, 8

[66] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Std: Sparse-to-dense 3d object detector for point cloud. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 2, 7

[67] Zetong Yang, Yin Zhou, Zhifeng Chen, and Jiquan Ngiam. 3d-man: 3d multi-frame attention network for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2

[68] Dongqiangzi Ye, Weijia Chen, Zixiang Zhou, Yufei Xie, Yu Wang, Panqu Wang, and Hassan Foroosh. Lidarmultinet: Unifying lidar semantic segmentation, 3d object detection, and panoptic segmentation in a single multi-task network. *CoRR*, abs/2206.11428, 2022. 6

[69] Maosheng Ye, Shuangjie Xu, and Tongyi Cao. Hvnet: Hybrid voxel network for lidar based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2

[70] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2, 6, 7

[71] Wenwei Zhang, Zhe Wang, and Chen Change Loy. Multi-modality cut and paste for 3d object detection. *CoRR*, abs/2012.12741, 2020. 5

[72] Yifan Zhang, Qingyong Hu, Guoquan Xu, Yanxin Ma, Jianwei Wan, and Yulan Guo. Not all points are equal: Learning highly efficient point-based detectors for 3d lidar point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2, 6, 7

[73] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H. S. Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 2

[74] Wu Zheng, Weiliang Tang, Sijin Chen, Li Jiang, and Chi-Wing Fu. Cia-ssd: Confident iou-aware single-stage object detector from point cloud. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021. 2

[75] Wu Zheng, Weiliang Tang, Li Jiang, and Chi-Wing Fu. Se-ssd: Self-ensembling single-stage object detector from point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2

[76] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2

[77] Zixiang Zhou, Xiangchen Zhao, Yu Wang, Panqu Wang, and Hassan Foroosh. Centerformer: Center-based transformer

for 3d object detection. In *Proceedings of the European Conference on Computer Vision*, 2022. 2

[78] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: deformable transformers for end-to-end object detection. In *International Conference on Learning Representations*, 2021. 2