Fully Sparse Transformer 3-D Detector for LiDAR Point Cloud

Diankun Zhang, Zhijie Zheng, Haoyu Niu, Xueqing Wang, and Xiaojun Liu, *Member, IEEE*

***Abstract*— The 3-D object detector usually uses a framework similar to 2-D detection and benefits from the advancements of 2-D detection tasks. In these frameworks, it is necessary to make the unstructured, sparse point cloud features into dense grids to be compatible with popular 2-D operators, such as convolution and transformers, which also causes extra computational costs. In this article, we propose a simple and efficient Fully Sparse TRansformer (FSTR) for light detection and ranging (LiDAR)-based 3-D object detection, which is able to combine with state-of-the-art (SOTA) sparse backbones to form a fully sparse, end-to-end, simple, and efficient detection framework. FSTR uses the sparse voxel feature from the sparse backbone as the input token without any custom operators. Furthermore, we introduce the dynamic queries (DQs) to provide a priori location and context of the foreground for the decoder and drop the high-confidence background tokens to further reduce redundant computations. We propose Gaussian denoising queries to speed up the decoder training and make it more adaptable to the distribution of sparse voxel features. Extensive experiments on the nuScenes benchmark and the Argoverse2 (AV2) benchmark validate the effectiveness of the proposed method. FSTR outperforms all LiDAR real-time methods by 69.5 mean average precision (mAP) and 72.9 nuScenes detection score (NDS) on the official benchmark of nuScenes dataset. On the long-range detection benchmark AV2, the proposed method achieves a new SOTA performance of 39.9 mAP, which outperforms the existing LiDAR detectors, even the LiDAR–camera detectors by a large margin (+9.4 and +7.5 mAP), showing the great advantage of the proposed method for long-range detection.**

***Index Terms*— 3-D detection, light detection and ranging (LiDAR), point cloud, sparse, transformer.**

I. Introduction

THE 3-D object detection is a key part of 3-D perception, and its performance has a significant impact on downstream tasks within the autonomous driving system [1], [2]. Light detection and ranging (LiDAR) point clouds are popularly used as input to 3-D detection networks, which are sparse, unstructured, and disordered. Voxelization is commonly used to structure the point cloud in order to use sparse convolutional networks for feature extraction (i.e., voxel-based method) [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], which has significant efficiency advantages compared with processing of sparse point clouds directly [13], [14], [15], [16], [17], [18]. Inspired by the anchor-based and center-based frameworks in 2-D detection [19], [20], 3-D anchor-based detectors and center-based detectors both have advanced performance [5], [11]. Dense convolution and dense feature maps are necessary for these detectors to generate dense predictions just like 2-D detection method and output the final results after nonmaximum suppression (NMS) postprocessing. On the one hand, the methods require a large number of handcrafted parameters (e.g., anchor and center mask) that hinder the performance of the detectors; on the other hand, these require converting the sparse feature maps output by sparse backbone [4] into dense feature maps and making dense predictions, resulting in unnecessary computational costs, a problem that is particularly evident in long-range detection tasks [21].

Fig. 1 visualizes the point cloud and corresponding feature maps on detection tasks with different ranges, i.e., nuScenes [22] and Argoverse2 (AV2) [21] from the commonly used submanifold convolution backbone [4], [23], [24]. The sparsity of voxel features increases with the detection range. The sparsity of the feature map in Fig. 1(a) is approximately 75% when the detection range is 50 m in nuScenes point clouds in Fig. 1(b). As the detection range is extended to 200 m in the AV2 dataset, as shown in Fig. 1(d), the sparsity of the feature map is more than 97% in Fig. 1(c). This means that most of the computations and memory are wasted with meanless in dense detection.

Recent works in 2-D detection have shown that the transformer [25] can avoid the hand-designed parameters in the detection pipeline and achieve end-to-end detection [26], [27], [28], [29]. Research on transformers in 3-D detection tasks has also been extensively carried out recently and achieved better performance [12], [30], [31], [32], [33]. However, the network computational burdensome or some complex operators are usually brought along with the transformers, which are detrimental to the deployment of the model in real applications, such as autonomous driving, that require concise, powerful, and real time. In recent years, many works have also tried to implement 3-D object detection directly using sparse point clouds or voxel features, but they usually have a very com-

Manuscript received 30 July 2023; revised 26 September 2023; accepted 27 October 2023. Date of publication 31 October 2023; date of current version 10 November 2023. This work was supported by the National Natural Science Foundation of China under Grant 61827803. *(Corresponding author: Xiaojun Liu.)*

Diankun Zhang, Zhijie Zheng, Haoyu Niu, and Xueqing Wang are with the Aerospace Information Research Institute and the Key Laboratory of Electromagnetic Radiation and Sensing Technology, Chinese Academy of Sciences, Beijing 100190, China, and also with the School of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: zhangdiankun19@mails.ucas.edu.cn; zhengzhijie19@mails.ucas.edu.cn; niuhaoyu21@mails.ucas.edu.cn; wangxueqing20@mails.ucas.edu.cn).

Xiaojun Liu is with the Aerospace Information Research Institute and the Key Laboratory of Electromagnetic Radiation and Sensing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: lxjdr@mail.ie.ac.cn).

Digital Object Identifier 10.1109/TGRS.2023.3328929

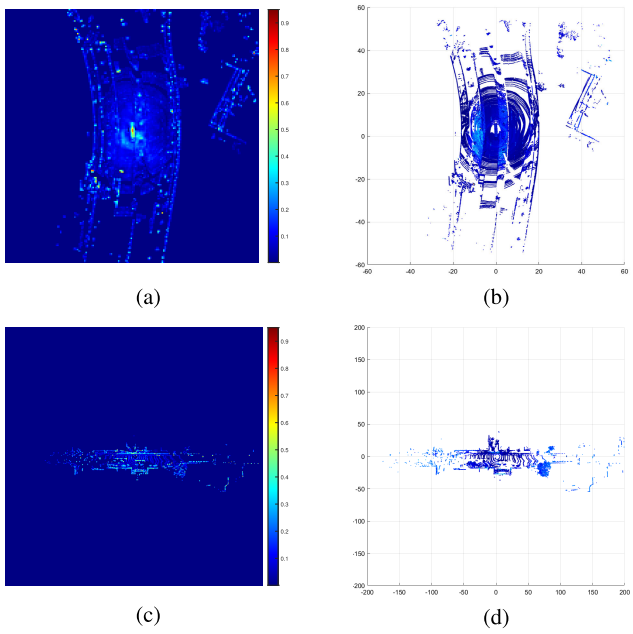

Fig. 1. Visualization of sparse BEV feature maps and point clouds with different ranges. (a) Sparse BEV feature map from nuScenes in range 50 m. (b) Raw point cloud from nuScenes in range 50 m. (c) Sparse BEV feature map from AV2 in range 200 m. (d) Raw point cloud from AV2 in range 200 m.

plex framework, and the efficiency or performance still lags behind the current optimal dense detectors [11], [16], [17], [18], [34], [35].

In this article, we propose Fully Sparse TRansformer (FSTR) for LiDAR-based 3-D detection. This is a simple, efficient, real-time, and powerful framework. By exploiting the permutation invariance of the transformer, FSTR takes the sparse BEV features as input tokens directly through BEV position encoding, which decouples the detection framework from dense BEV features and allows the network to be combined with any advanced 3-D point cloud (sparse) backbone. Furthermore, along with learnable query with the statistic-level prior, we use a coarse lightweight sparse detection head to provide the transformer with a sample-level priori position information and prior context feature about the object, i.e., dynamic query (DQ). The proposed DQ significantly reduces the difficulty of queries in decoders to search for foreground targets within the global receptive field; thus, the number of the learnable queries and the decoders can be significantly reduced to improve the efficiency as well as the performance of the model. FSTR also drops the high-confidence background tokens to further reduce redundant computations. We further propose the Gaussian denoising query (GDQ) for sparse 3-D features to further promote the decoder to adapt to the distribution of sparse voxel features and the DQs.

To summarize, our contributions are listed as follows.

1) We propose the simple and efficient FSTR, which can be directly applied to the existing 3-D sparse/dense LiDAR backbone with real-time and strong detection performance, especially for long-range detection.
2) We propose DQs for FSTR, which provides sample-level prior position information and context feature for the foreground object. The DQs greatly reduce the difficulty of searching the foreground compared with the commonly used learnable queries and significantly improve the performance and efficiency of the detectors. FSTR also drops the high-confidence background tokens to further reduce redundant computations.
3) We propose the Gaussian denoising queries to further promote the decoder to adapt to the distribution of sparse features.
4) As a real-time detector, FSTR outperforms all existing fully sparse 3-D LiDAR detection frameworks and sets new records on the challenging nuScenes detection benchmark, as well as the existing state-of-the-art (SOTA) methods on the large-scale long-range AV2 detection benchmark by a large margin.

II. Related Works

A. LiDAR-Based 3-D Object Detection

Due to the sparse and unstructured nature of LiDAR point clouds, localization and classification of 3-D targets remain a challenging task. Current 3-D detectors can be divided into two main frameworks, point-based or voxel-based paradigms, respectively.

Point-based methods process point clouds as raw points. PointNet [13], PointNet++ [14], or their variants [37], [38], [39] usually are token as the backbone to extract point-level features from point clouds to keep the order invariance. For example, point region-convolutional neural network (Point-RCNN) [16] is a typical two-stage method that uses point-level features for segmentation and obtains region of interest (ROI) from the foreground points for final prediction. Point-GNN [40] proposes a graph neural network for 3-D detection. VoteNet [41] is a one-stage point-based 3-D detector where deep Hough voting is leveraged to regress the instance centroid. Although point-based methods take full advantage of the sparsity of point clouds, the computational cost of these methods, which positively related to the number of points, makes they are too heavy to perceptron the large-scale scene in autonomous driving. Downsampling is usually performed by farthest point sampling (FPS) or its variants [42], [43] to alleviate the latency, which inevitably results in information loss. At the same time, due to the unstructured nature of point clouds, point-based methods require frequent nearest neighbor searches in 3-D continuous space for local information aggregation and target assignment [14], [40], [44], [45], which also causes inefficiency of the methods. The unstructured nature also makes it necessary to have a specially designed backbone network to extract features, which often leads to insufficient learning capability of the network. Overall, point-based methods currently lag behind voxel-based methods in terms of efficiency and performance.

Voxel-based detectors usually divide unstructured point clouds into regular voxel (or pillar) grids in order to extract features using efficient convolution operations as 2-D images [19], [46]. VoxelNet [3] uses 3-D convolution for feature extraction of dense voxels, which leads to unnecessary computational costs. SECOND [4] proposes a sparse backbone consisting of sparse convolution and submanifold convolution [23], [24], which greatly improves the efficiency of the

network, and its backbone has been widely used in subsequent studies. PointPillar [6] further simplifies the voxels into pillars to realize the lightweight of the network. PillarNet [35] achieves a better performance-efficiency trade-off by a larger encoder based on the framework of PointPillar. CenterPoint [11] extends CenterNet [20] and CornerNet [47] into 3-D detection task and achieves high performance with a simple framework. FocalConv [48] further optimizes the sparse convolution operator for a more flexible receptive field. Single-stride sparse transformer (SST) [12] utilizes the popular all-transformer backbone in 3-D object detection to solve the difficulties in small object detection. In summary, the voxel-based methods usually have good efficiency and detection performance benefit from the high-efficiency and powerful sparse convolution backbone but also introduce potential information loss in voxelization.

Although voxel-based methods are already efficient, they often still rely on the dense prediction base on the dense bird's eyes view (BEV) features, which is often achieved by a 2-D dense convolution backbone and a dense prediction head [4], [11]. The dense convolutions effectively alleviate the insufficient receptive fields of sparse backbone, but also brings a lot of unnecessary computation. The studies of fully sparse 3-D detectors have received increasing attention in recent years. Inspired by VoteNet [41], fully sparse detector (FSD) [49] uses point clustering and group correction to solve the problem of missing object centroids on sparse feature maps. VoxelNeXt [36] proposes a powerful sparse backbone structure to alleviate the problem of insufficient perceptual field of the submanifold backbone and achieve a better speed–accuracy trade-off. However, its performance is still bottlenecked by the receptive field and at a disadvantage compared with the SOTA 3-D detectors.

B. Transformer-Based Object Detection

Transformer-based detectors are well studied for the past few years, and detection transformer (DETR) [26] first proposed the transformer framework for 2-D end-to-end detection tasks, avoiding the complex handcraft hyperparameter of the detection framework and achieving SOTA performance. Many improvement works based on DETR have been proposed to improve the performance of DETR through better query design, more efficient transformer layers, and more advanced training methods, and dominate the 2-D detection field in popular benchmarks [27], [28], [29], [50], [51], [52]. Deformable-DETR [27] proposes deformable attention that only focuses on the region around the reference points, which greatly reduces the computational cost and makes it more flexible to introduce multiscale features in the transformer. Denoising DETR (DN-DETR) [28] and DETR with improved deNoising anchOr (DINO) [50] propose the denoising query that further accelerates the convergence of the model. Group-DETR [51] and selective query recollection (SQR) [52] further investigate the effect of different supervision on the performance and convergence speed of transformer-based detectors.

In the last few years, there are many transformer-based detection frameworks proposed for 3-D detection tasks [12], [30], [32], [33], [53], [54]. SST [12] is inspired by Swin [55] and proposes a self-attentive encoder for sparse features.

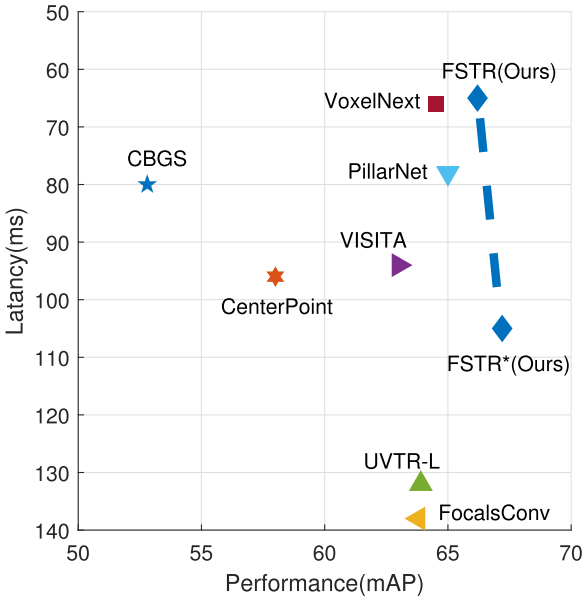

Fig. 2. Compared with other 3-D object detectors, the proposed FSTR achieves better performance-lantency trade-off. The results are evaluated on nuScenes [22] test set. *: FSTR with a double-channel sparse backbone proposed in [36].

Tranfusion [33] aggregates information from LiDAR and camera by the deformable transformer. Conquer [30] introduces contrast learning to achieve effective suppression of false positive prediction. Cross-modal transformer (CMT) [32] elaborates the position embedding of LiDAR and camera and uses global cross attention to aggregation of information. Compared with other transformer-based detectors, CMT has a more concise structure, but global attention makes its head too heavy, especially for high-resolution LiDAR BEV features. Inspired by CMT, we propose FSTR, an FSTR detection framework that achieves SOTA performance of the fully sparse framework detector with real-time detection capability.

III. Mehtod

In this article, we propose the FSTR to achieve a better efficiency–accuracy trade-off in 3-D detection tasks. Fig. 2 shows the comparison of the performance and inference latency between the proposed method and the previous works. The framework of FSTR is shown in Fig. 3, which contains four main parts: DQ generator (Section III-B), sparse context assigner (Section III-C), and sparse transformer (ST) decoder (Section III-D). Inspired by CMT and VoxelNeXt, we also introduce Gaussian denoising queries (Section III-E) to further improve the training quality of the network.

A. Problem Definition

Similar to the popular voxel-based detector, a sparse convolutional backbone is used to extract features from the point cloud, such as SECOND [4] or VoxelNeXt [36], and outputs sparse voxel features on the BEV. The sparse voxels are represented as a feature vector and a BEV index vector. For the sparse voxel $\mathcal{V} = \{\mathbf{v}_k; k = 1, \ldots, N\}$ output by the sparse backbone, we denote its feature vector as $\mathcal{F} = \{\mathbf{f}_i; i = 1, \ldots, N\}$ and the BEV index vector as $\mathcal{I} = \{\mathbf{i}_k; k = 1, \ldots, N\}$. By normalizing the index vector, we can obtain the BEV position vector of the voxel $\mathcal{P} = \{\mathbf{p}_k; k = 1, \ldots, N\}$. N is the number of sparse voxels, which varies with the

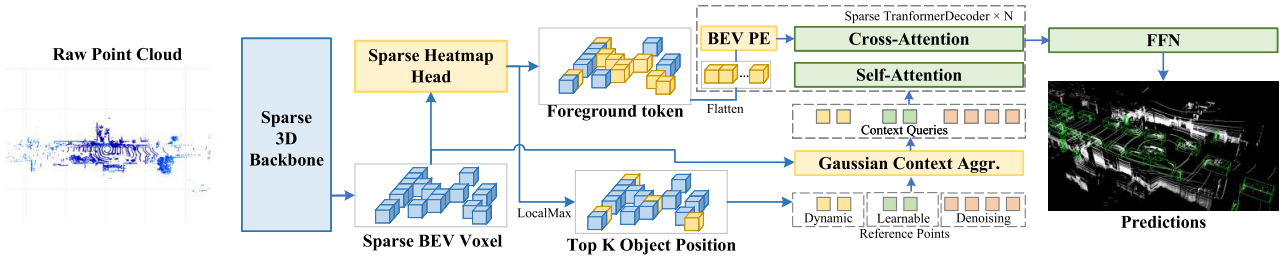

Fig. 3. Simple and efficient architecture of the FSTR paradigm. The raw point cloud is fed into a sparse 3-D backbone network and output as a sparse BEV feature map. Foreground scores are first predicted for these sparse voxel features by a sparse heat-map head. The foreground scores are used to not only filter a fixed number of foreground voxels as input tokens for the decoder but also provide coarse locations of interested objects after local maxima suppression. The DQs are set on these coarse locations as input to the decoder together with the learnable queries. We proposed Gaussian denoising queries to further accelerate model convergence under sparse input. The Gaussian context aggregator (GCA) is proposed to aggregate context features for dynamic, learnable, and denoising queries. The foreground tokens and queries are encoded using their positions in the BEV space and input to the decoder to predict 3-D bounding boxes.

point cloud in each sample. During the training, we denote ground-truth bounding boxes by $\mathcal{B} = \{\mathbf{b}_j; j = 1, \ldots, M\}$, and $\mathcal{C} = \{\mathbf{c}_j; j = 1, \ldots, M\}$ denotes the center position vector of ground-truth bounding boxes $\mathcal{B}$.

B. DQ Generator

Transformer-based detectors are usually equipped with learnable queries, however, since learnable queries can only rely on the prior distribution learned from training data to determine the distribution of queries in space, leading to greater difficulty in finding targets during inference [56]. In addition, such methods generally scatter a large number of queries in the interested space to guarantee the recall of detection. Since the interested space (BEV) grows squarely with the detection range in the 3-D detection task, the number of queries increased dramatically. For example, when the detection range is enlarged from 50 m (nuScenes) to 200 m (AV2), 16× queries are needed to ensure the same distribution density in the whole space, which are usually unacceptable computational cost. In this article, we propose the DQ to fully leverage the sample-level information to alleviate the redundancy of learnable queries with only dataset-level prior. The DQs are settled on the most potential foreground position and dynamic change for each sample. The DQ generator uses a lightweight heat-map head to give a score for each voxel selects top-K potential foreground voxels as the rough object prior and sets a DQ at this position.

For sparse voxels $\mathbf{v}_k \in \mathcal{V}$, the heat-map score $\mathbf{s}_k \in \mathcal{S}$ can be obtained as follows:

$$\mathbf{s}_k = H(\mathbf{v}_k) \tag{1}$$

where H is a heat-map head that uses submanifold convolution instead of regular convolution [11], [36] and $\mathbf{s}_k \in \mathbb{R}^C$, where C is the number of classes. The target score $\mathbf{t}_k$ is assigned for each sparse voxel $\mathbf{v}_k$ according to its position vector $\mathbf{p}_k$ by two Gaussian weights, and the target score $\mathbf{t}_k$ assigned to voxel $\mathbf{v}_k$ for each ground-truth center $\mathbf{c}_n$ for position $\mathbf{p}_k$ is obtained as follows:

$$t_{k,l} = \min\left(\exp\left(\frac{|\mathbf{p}_k - \mathbf{c}_n|}{\frac{1}{6}(2r+1)}\right) + \exp\left(\frac{-|\mathbf{p}_k - \mathbf{p}_{\text{nearest},\mathbf{c}_n}|}{\frac{1}{6}(2r+1)}\right), 1\right) \tag{2}$$

where $\mathbf{p}_{\text{nearest},\mathbf{c}_n} = \arg\min_{\mathbf{p}_j \in \mathcal{P}} |\mathbf{p}_j - \mathbf{c}_n|$ is the position vector of the nearest sparse voxel to the ground-truth center position, l is the class of ground-truth bounding boxes $\mathbf{b}_n$ centering on center vector $\mathbf{c}_n$, and r is a hyperparameter, which we take $r = 2$ in our experiments. The center of the first Gaussian distribution lies at the center $\mathbf{c}_n$ of the ground-truth bounding box $\mathbf{b}_n$, which is the same as the dense heat-map prediction. However, sparse voxel features often meet the problem of missing central features due to the absence of dense convolution to dilate the feature from the edges to the center of objects. To ensure that each ground-truth bounding box $\mathbf{b}_n$ is assigned to at least one voxel $\mathbf{v}_k$, the second Gaussian distribution is located at the position of the sparse voxel closest to the ground-truth center $\mathbf{c}_n$. The values of the two Gaussian weights are summed as the target scores of sparse voxels after truncated by 1, as shown in (2).

The above assignment may lead to a one-to-many assign result, making the prediction with multiple false positive predictions around the ground truth. Local maxima suppression can effectively suppress this situation, as follows:

$$\mathcal{S}_l = \{\mathbf{s} * I(\mathbf{s} = \text{sparse\_maxpool}(\mathbf{s}, \mathbf{p}, \mathcal{S}, \mathcal{P})) \mid \mathbf{s} \in \mathcal{S}, \mathbf{p} \in \mathcal{P}\}. \tag{3}$$

Note that here the local maxima suppression is conducted independently for each class. For some dense small objects (e.g., pedestrians, cones, and bollards), it is not necessary to use local maxima suppression. Finally, we set the DQ on the voxel position with the top-K score from $\mathcal{S}_l$. We take $K = 200$ in our experiments.

C. Sparse Context Assigner

The sparse context assigner proposed in this section initializes the context query feature according to the reference point of the query. Traditional methods generally use bilinear interpolation to obtain features at arbitrary locations from dense BEV feature maps; however, this is not suitable for sparse BEV feature maps. As shown in Fig. 3, the query of the proposed method consists of three parts, DQs, learnable queries, and denoising queries (only training). Among them, the positions of DQs are definitely on the sparse voxel positions, while the others are not. Therefore, this article proposes a Gaussian-weighted sparse context assigner to aggregate the

context feature on the sparse feature map. The context feature of a query with reference points at position $\mathbf{r}$ can be expressed as follows:

$$\mathbf{q}(\mathbf{r}) = \frac{\sum_{\mathbf{p},\mathbf{f}\in\mathcal{N}(\mathcal{P},\mathcal{F})} \exp\left(\frac{-|\mathbf{r}-\mathbf{p}|}{w+\sigma}\right)\cdot \mathbf{f}}{\sum_{\mathbf{p}\in\mathcal{N}(\mathcal{P})} \exp\left(\frac{-|\mathbf{r}-\mathbf{p}|}{w+\sigma}\right)} \tag{4}$$

where w is the decay factor to control the range of the context feature that queries aggregate, σ is a constant factor, and $\mathcal{N}$ is a neighborhood, which centered at $\mathbf{r}$. In particular, when the query directly uses the sparse voxel feature (zeros if voxel is empty) as the context feature, it can be regarded as $w(\mathbf{p},\mathbf{r}) = \delta(\mathbf{p}-\mathbf{r})$; when the query uses its closest sparse voxel feature as the context feature, it can be regarded as $w(\mathbf{p},\mathbf{r}) = \delta(\mathbf{p}-\mathbf{p}_{N(\mathbf{r}))}$, where $\mathbf{p}_{N(\mathbf{r}_p))}$ is the closest voxel position with the reference point $\mathbf{r}$ and δ is the Dirac function satisfying

$$\delta(x) = \begin{cases} 1, & x = 0 \\ 0, & x \neq 0. \end{cases} \tag{5}$$

D. ST Decoder

The proposed ST uses the sparse voxel feature extracted by the sparse backbone as the input token and uses the query context feature obtained in (2) as the initialization context feature of the query. The reference points of query and token positions are normalized to BEV coordinates. For BEV continuous coordinates (u, v), we use sinusoid position encoding [25] followed by a learnable feedforward neural network (FFN) as their position encoding for both query and token as follows:

$$\text{BEVPE} = \text{FFN}(\text{concat}(\text{PE}(u), \text{PE}(v))) \tag{6}$$

where

$$\text{PE}(x)_{2i} = \sin\left(x/10000^{2i/d}\right) \tag{7}$$

$$\text{PE}(x)_{2i+1} = \cos\left(x/10000^{2(i+1)/d}\right). \tag{8}$$

Notice that the decoder performs cross attention on the global receptive field, which alleviates the problem of insufficient receptive field for the sparse backbone. But, the global cross attention also brings some unnecessary computations, such as the interaction of query with a large number of background tokens. Similar to the DQ generator, we use the heat-map score of the sparse voxel as the metric for selecting valuable tokens and select the top-M tokens for input into the decoder. The token selecting processing reduces the number of tokens by about 70% on the nuScenes dataset (typically more than 30 000 tokens) and 96% on the AV2 dataset (typically more than 260 000 tokens) compared with using the dense BEV feature as input tokens.

E. Gaussian Denoising Queries

Enhancing the training quality of the decoder improves the training efficiency and the performance of the model. Based on DN-DETR [28], CMT [32] introduces an additional noise threshold to divide the denoising query into positive and negative samples. However, we find that the existing noise distribution of the denoising query may not be optimal for DQs in the proposed fully sparse method. Inspired by VoxelNeXt [36], Table I shows the position distribution of DQ relative to ground-truth bounding box centers in the proposed method. Due to the missing central feature problem of the sparse backbone, the DQ is closer to a Gaussian distribution centered on the object boundary rather than a uniform distribution centered on the center of ground-truth bounding box centers. In the existing methods [32], the probability of denoising query distributing on the boundary is much smaller and will be classified as a negative sample, which is gapped from the actual inference of the model. In this article, we propose Gaussian denoising queries, which use a larger noise threshold based on the characteristics of the sparse voxel feature and use Gaussian distribution instead of the uniform distribution to sample the noise. As shown in Fig. 4. The noise amplitude is obtained by sampling a Gaussian distribution, such that the denoising query is more likely to be distributed near the edges, and a larger noise threshold is used to involve the denoising query around the edges into the positive samples. Obviously, this distribution is closer to the actual situation in the proposed sparse method that most of the DQs are set around the edge of the ground truth to regress to the bounding boxes, as shown in Table I.

TABLE I
PROPORTION OF THE POSITIONAL DISTRIBUTION OF DQ RELATIVE TO GROUND-TRUTH BOUNDING BOXES WHEN THEY ARE MATCHED DURING TARGET ASSIGNING

	Near center	Near boundary	Outside box
Proportion	12.4%	70.7%	16.9%

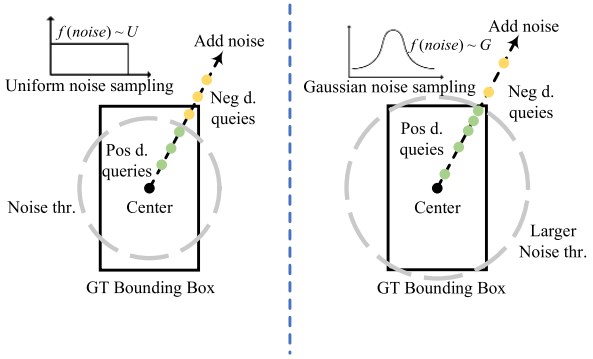

Fig. 4. Schematic of the proposed GDQ. Left: denoising methods proposed in CMT [32]. Right: Gaussian denoising queries proposed in this article, which aims to accelerate the convergence of the decoder under sparse inputs. The method concentrates the denoising queries more on the edges of the ground-truth bounding box and uses a larger noise threshold to more closely approximate the queries distribution during inference. Thr.: threshold. D.: denoising.

IV. EXPERIMENTS

A. Datasets and Evaluation Metrics

We use the popular nuScenes dataset [22] and the large-scale long-range AV2 dataset [21] to evaluate the performance of our method.

TABLE II

PERFORMANCE OF 3-D OBJECT DETECTION METHODS ON NUSCENES TEST SET. ‡: RESULTS ARE EVALUATED WITH TTA. ALL METHODS LISTED IN THIS TABLE TAKE LIDAR POINT CLOUD AS ONLY MODAL INPUT. FSTR-L: PROPOSED METHOD WITH A DOUBLE-CHANNEL BACKBONE. FSTR-XL: PROPOSED METHOD WITH A LARGER DOUBLE-CHANNEL RESNET-34-LIKE VOXELNEXT BACKBONE AND SMALLER INPUT VOXEL SIZE [0.05, 0.05, 0.2]

Method	mAP	NDS	Latency	mAP per Class									
				Car	Truck	Bus	Trailer	C.V.	Ped	Mot	Byc	T.C.	Bar
PointPillars [6]	30.5	45.3	31 ms	68.4	23.0	28.2	23.4	4.1	59.7	27.4	1.1	30.8	38.9
3DSSD [42]	42.6	56.4	-	81.2	47.2	61.4	30.5	12.6	70.2	36.0	8.6	31.1	47.9
CBGS [57]	52.8	63.3	80 ms	81.1	48.5	54.9	42.9	10.5	80.1	51.5	22.3	70.9	65.7
CenterPoint [11]	58.0	65.5	96 ms	84.6	51.0	60.2	53.2	17.5	83.4	53.7	28.7	76.7	70.9
HotSpotNet [58]	59.3	66.0	-	83.1	50.9	56.4	53.3	23.0	81.3	63.5	36.6	73.0	71.6
AFDetV2 [59]	62.4	68.5	-	86.3	54.2	62.5	58.9	26.7	85.8	63.8	34.3	80.1	71.0
Focals Conv [48]	63.8	70.0	138 ms	86.7	56.3	67.7	59.5	23.8	87.5	64.5	36.3	81.4	74.1
VISTA‡ [60]	63.0	69.8	94 ms	84.4	55.1	63.7	54.2	25.1	82.8	70.0	45.4	78.5	71.4
UVTR-L‡ [61]	63.9	69.7	132 ms	86.3	52.2	62.8	59.7	33.7	84.5	68.8	41.1	74.7	74.9
PillarNet-18‡ [35]	65.0	70.8	78 ms	87.4	56.7	60.9	61.8	30.4	87.2	67.4	40.3	82.1	76.0
VoxelNeXt [36]	64.5	70.0	66 ms	84.6	53.0	64.7	55.8	28.7	85.8	73.2	45.7	79.0	74.6
Transfusion-L [33]	65.5	70.2	-	86.2	56.7	66.3	58.8	28.2	86.1	68.3	44.2	82.0	78.2
LargeKernel [62]	65.3	70.5	-	85.9	55.3	66.2	60.2	26.8	85.6	72.5	46.6	80.0	74.3
FSTR (ours)	66.2	70.4	65ms	85.8	53.9	64.1	57.4	31.1	87.5	74.1	48.2	81.4	78.1
FSTR-L (ours)	67.2	71.5	105ms	86.5	54.1	66.4	58.4	33.4	88.6	73.7	48.1	84.4	78.1
FSTR-L‡(ours)	69.5	72.9	-	87.2	57.0	69.3	61.0	35.7	89.8	77.9	52.8	84.0	79.8
FSTR-XL‡(ours)	**70.2**	**73.6**	-	87.1	58.4	69.3	60.6	32.9	90.0	80.2	57.0	85.7	80.6

TABLE III

PERFORMANCE OF 3-D OBJECT DETECTION METHODS ON NUSCENES VALIDATION SET. ALL METHODS ARE EVALUATED WITHOUT ANY TTA OR MODEL ENSEMBLE

Method	mAP	NDS
CBGS [57]	51.4	62.6
SECOND [4]	52.6	63.0
CenterPoint [11]	59.6	66.8
FSD [49]	62.5	68.7
VoxelNeXt [36]	63.5	68.7
LargeKernel3D [62]	63.3	69.1
Focals Conv [48]	61.2	68.1
FSTR (Ours)	64.4	69.3
FSTR-L (Ours)	**65.5**	**70.3**

NuScenes [22] consists of 28 130 training samples and 6019 validation samples. Each sample is annotated in ten categories with 3-D bounding boxes. The point clouds in nuScenes dataset are collected by a single 32-beams LiDAR sensor, which has a detection range of approximately 60 m. According to the nuScenes metrics, we adopt the five types of true positive (TP) metrics, including average translation error (ATE), average scale error (ASE), average orientation error (AOE), average velocity error (AVE), and average attribute error (AAE) for measuring qualities of TP prediction. We also conduct experiments on the dataset and compare the results with other methods using the following commonly used mean average precision (mAP) and the nuScenes detection score (NDS).

AV2 [21] is a large-scale long-range perceptron dataset for autonomous driving. AV2 contains 1000 scenes in total, and these total scenes are divided into 700 for training, 150 for validation, and 150 for testing. The 3-D bounding boxes in this dataset are annotated in 26 categories with a long-tail form. The metrics is similar to the metrics of nuScenes, which also includes three types of TP metrics; ATE, ASE, and AOE; the commonly used metric mAP; and the composite detection score (CDS), which combines all the above metrics. The perception range in AV2 reaches 200 m (total area 400×400 m), which is much larger than nuScenes. Such a large perception range leads to a huge memory and computational cost for dense detectors. We use the AV2 dataset to further evaluate the performance and efficiency of the model on long-range detection tasks.

B. Experimental Setup

1) Input Parameters: For the nuScenes dataset, the detection range is set to be $[-54, 54]$ m for both the X- and Y-axis and $[-5, 3]$ m for the Z-axis. The voxel size is set to (0.075, 0.075, and 0.1) m. For the AV2 dataset, the detection range is set to be $[-204.8, 204.8]$ m for both the X- and Y-axis and $[-4, 4]$ m for the Z-axis. The voxel size is set to (0.1, 0.1, and 0.2) m due to the large range of input data.

2) Implementation Details: The following experiments are performed on the Ubuntu 20.04 operating system with eight pieces of GPUs of NVIDIA RTX 3090 24 GB and dual Intel Xeon Gold 5318Y CPU processors with 2.10-GHz frequency. The network training and testing processes are implemented by using Python 3.8 language environment and PyTorch 1.8.1.

TABLE IV

COMPARISON WITH STATE-OF-THE-ART METHODS ON AV2 VALIDATION SPLIT. C-BARREL: CONSTRUCTION BARREL. MPC-SIGN: MOBILE PEDESTRIAN CROSSING SIGN. A-BUS: ARTICULATED BUS. C-CONE: CONSTRUCTION CONE. V-TRAILER: VEHICULAR TRAILER. †: PROVIDED BY AUTHORS OF AV2 DATASET. ‡: REIMPLEMENTED BY US. *: REIMPLEMENTED BY FSD [49]. SOME CATEGORIES ARE EXCLUDED FROM THE TABLE DUE TO THE LIMITED NUMBER OF INSTANCES THEY CONTAIN. HOWEVER, THE AVERAGE RESULTS CONSIDER ALL CATEGORIES, EVEN THOSE THAT ARE OMITTED

	Methods	Average	Vehicle	Bus	Pedestrian	Box Truck	C-Barrel	Motorcyclist	MPC-Sign	Motorcycle	Bicycle	A-Bus	School Bus	Truck Cab	C-Cone	V-Trailer	Bollard	Sign	Large Vehicle	Stop Sign	Stroller	Bicyclist
mAP	CenterPoint† [11]	13.5	61.0	36.0	33.0	26.0	22.5	16.0	16.0	12.5	9.5	8.5	7.5	8.0	8.0	7.0	25.0	6.5	3.0	28.0	2.0	14
	CenterPoint* [11]	22.0	67.6	38.9	46.5	40.1	32.2	28.6	27.4	33.4	24.5	8.7	25.8	22.6	29.5	22.4	37.4	6.3	3.9	16.9	0.5	20.1
	CenterPoint‡ [11]	27.4	78.7	45.3	62.2	43.7	59.3	10.3	45.4	33.2	30.6	25.8	27.2	23.2	36.7	23.8	50.3	15.0	4.9	47.9	0.2	15.2
	FSD [49]	28.2	68.1	40.9	59.0	38.5	42.6	39.7	26.2	**49.0**	38.6	20.4	30.5	14.8	41.2	26.9	41.8	11.9	5.9	29.0	13.8	33.4
	VoxelNeXt [36]	30.5	72.0	39.7	63.2	39.7	64.5	46.0	34.8	44.9	40.7	21.0	27.0	18.4	44.5	22.2	53.7	15.6	7.3	40.1	11.1	34.9
	FSF† [63]	33.2	70.8	44.1	60.8	40.2	50.9	48.9	28.3	60.9	47.6	22.7	36.1	26.7	51.7	28.1	41.1	12.2	6.8	27.7	25.0	41.6
	FSTR(Ours)	38.3	77.3	46.3	70.3	45.4	**72.3**	56.4	54.1	43.5	45.1	25.8	45.1	28.4	55.3	**31.8**	61.8	21.9	8.6	46.4	30.0	44.8
	FSTR-L(Ours)	**39.9**	**78.7**	**48.2**	**72.3**	**46.1**	71.0	**59.8**	**56.7**	45.7	**50.8**	**29.6**	**45.5**	**29.2**	**58.3**	31.2	**62.7**	**22.1**	**9.2**	**49.7**	**34.1**	**48.8**
CDS	CenterPoint* [11]	17.6	57.2	32.0	35.7	31.0	25.6	22.2	19.1	28.2	19.6	6.8	22.5	17.4	22.4	17.2	28.9	4.8	3.0	13.2	0.4	16.7
	CenterPoint‡ [11]	21.0	66.5	37.1	45.1	35.6	45.5	7.0	31.0	24.0	22.7	20.8	22.6	17.8	26.2	17.4	35.4	11.1	3.5	37.8	0.2	11.6
	FSD [49]	22.7	57.7	34.2	47.5	31.7	34.4	32.3	18.0	**41.4**	32.0	15.9	26.1	11.0	30.7	20.5	30.9	9.5	4.4	23.4	11.5	28.0
	VoxelNeXt [36]	23.0	57.7	30.3	45.5	31.6	50.5	33.8	25.1	34.3	30.5	15.5	22.2	13.6	32.5	15.1	38.4	11.8	5.2	30.0	8.9	25.7
	FSF† [63]	25.5	59.6	35.6	48.5	32.1	40.1	35.9	19.1	48.9	37.2	17.2	29.5	19.6	37.3	21.0	29.9	9.2	4.9	21.8	18.5	32.0
	FSTR(Ours)	30.0	65.6	37.9	54.4	36.9	**57.0**	46.6	37.1	34.6	36.0	20.7	37.8	21.7	40.7	**24.2**	44.9	17.1	6.4	37.4	22.8	36.2
	FSTR-L(Ours)	**31.5**	**67.0**	**39.6**	**56.5**	**37.8**	55.1	**49.7**	**39.7**	36.7	**41.2**	**24.3**	**38.3**	**22.4**	**42.7**	24.0	**45.4**	**17.5**	**6.8**	**40.2**	**26.5**	**39.6**

The runtime analysis of the proposed method is performed with a single NVIDIA RTX 3090 24-GB GPU.

Our model is a quite simple framework that is only composed of a sparse 3-D backbone and a 3-D detection head with a single decoder. Benefiting from high compatibility with the sparse backbone of FSTR, we adopt advanced VoxelNeXt [36] as our sparse backbone to conduct main experiments. In the detection head, we use a total of 500 queries and divide them into 200 DQs and 300 learnable queries for better recall. For a better efficiency–performance trade-off, we use only one decoder in FSTR and only take 10 000 tokens from the sparse BEV feature map as the input of the cross-attention module. In addition, we use flash attention [64] to replace the multihead attention [25] provided in PyTorch [65] to further improve the model efficiency during the training process. These configurations are the same for both the nuScenes and AV2 datasets. We use a 3×3 sparse max pooling for local maxima suppression to filter duplicate DQs around potential foreground objects. However, for some small objects that are densely occurring, the recall will drop significantly by this operation. Therefore, we use dummy operation (max pooling with 1×1 kernel) for *pedestrian* and *traffic cone* in the nuScenes dataset, and *bicycle*, *bicyclist*, *bollard*, *construction cone*, and *pedestrian* in the AV2 dataset.

3) Training and Inference Details: Our FSTR method is end-to-end optimized with the Adam optimizer, 0.01 weight decay, and a base learning rate of 0.0001. All experiments are performed at FP16 precision. We train the model for 20 epochs with class-balanced grouping and sampling (CBGS) [57] on the nuScenes dataset [22] with a batch size of 4, and 20 epochs without CBGS on the AV2 [21] with a batch size of 4. We adopt data augmentation strategies as common settings for both datasets, which include global random flipping about the X- or Y-axis with a probability of 0.5, respectively, global scaling with a random scaling factor between 0.95 and 1.05, global rotation about the Z-axis with a random angle factor between $-\frac{1}{4}\pi$ and $\frac{1}{4}\pi$, and ground-truth augmentation (GT-AUG) [4] to alleviate the category imbalances.

During inference, the proposals with top-200 scores will be output as the final predictions without NMS in postprocessing.

C. Benchmarking Result

In Table II, the proposed method is evaluated on the nuScenes test split by the official benchmark and compared with other SOTA methods. The methods marked with ‡ [35], [36], [60], [61] are reported with the double-flip testing augmentation. FSTR achieves leading performance compared with the SOTA methods in Table II. Our real-time detector outperforms other methods by +0.9 mAP and with only 65-ms inference latency. With a large backbone (double the channels), FSTR achieves 67.2 mAP and 71.5 NDS in almost real-time latency (105 ms). With the test time augmentations (TTAs), the proposed method reaches 69.5 mAP and 72.9 NDS on the nuScenes test set. Benefiting from the larger receptive field of global attention, FSTR has shown great advantages for small object detection, such as *pedestrian, motorcycle, bicycle*, and *barrier*. We further explore the performance of the proposed method with a scale-up backbone. With scale-up VoxelNeXt backbone (ResNet34-like block setting [66] and double channel), smaller input voxel size ([0.05, 0.05, 0.2]), and TTAs, the proposed method reaches 70.2 mAP and 73.6 NDS on the nuScenes test set, which are much higher than the other methods listed in Table II. We also report the evaluation results of the proposed method on the validation samples of the challenge nuScenes dataset in Table III. FSTR also achieves competitive results on both mAP and NDS metrics compared with previous methods.

We also report the evaluation results of the FSTR method on the 150 validation sequences of large-scale long-range AV2 dataset in Table IV. FSTR outperforms the previous LiDAR-based detection method [11], [36], [49] by +9.4 mAP and +7.5 CDS on the average metrics, even great better than the SOTA fusion detection method [63] by +6.7 mAP and +6.0 CDS on the average metrics. We show the per-class metric in Table IV, which shows that FSTR has a significant and consistent advantage in all classes. We believe is due to

TABLE V

RESULTS EVALUATED OVER DIFFERENT RANGES ON AV2 DATASET AND COMPARED WITH THE STATE-OF-THE-ART METHOD TO FURTHER VALIDATE THE PERFORMANCE OF DIFFERENT METHODS AT DIFFERENT DISTANCES IN MORE DETAIL. ‡: REIMPLEMENTED BY US. THE AVERAGE RESULTS CONSIDER ALL CATEGORIES, EVEN THOSE THAT ARE OMITTED IN THIS TABLE

Range		Methods	Average	Vehicle	Bus	Pedestrian	Box Truck	C-Barrel	Motorcyclist	MPC-Sign	Motorcycle	Bicycle	A-Bus	School Bus	Truck Cab	C-Cone	V-Trailer	Bollard	Sign	Large Vehicle	Stop Sign	Stroller	Bicyclist
0-50m	mAP	CenterPoint‡	35.6	90.1	63.2	73.8	64.8	74.6	12.2	59.6	43.7	38.6	38.7	47.2	29.7	46.0	30.5	57.4	19.1	7.5	62.1	0.5	26.3
		FSD‡	44.0	88.5	67.4	77.2	67.2	81.0	70.8	70.1	**64.3**	50.8	21.7	61.8	9.9	49.9	35.4	58.3	25.0	14.6	51.5	33.2	65.7
		VoxelNeXt‡	40.4	87.4	61.7	75.7	60.5	77.4	56.1	63.6	51.3	41.6	31.4	55.3	27.7	52.6	31.9	60.8	21.7	11.4	61.0	10.3	52.1
		FSTR(Ours)	**52.6**	**91.5**	**72.5**	**84.2**	**72.2**	**83.4**	**77.4**	**81.0**	54.5	**57.2**	**46.3**	**82.0**	**40.7**	**69.7**	**44.9**	**73.5**	**27.5**	**17.1**	**67.3**	**37.8**	**73.8**
	CDS	CenterPoint‡	28.0	79.9	53.9	55.2	54.8	57.4	8.1	43.1	31.9	29.1	32.0	39.8	23.3	33.4	22.8	41.1	14.0	5.6	49.4	0.3	20.5
		FSD‡	36.0	77.9	57.7	63.0	57.1	66.4	60.5	49.1	55.3	42.8	16.3	**53.3**	7.1	37.6	26.3	43.9	20.3	11.2	42.5	27.2	56.6
		VoxelNeXt‡	31.5	77.2	52.4	57.8	50.5	59.2	46.4	42.9	37.2	31.5	24.2	46.1	20.3	38.5	23.7	43.6	15.9	8.7	47.9	7.4	42.2
		FSTR(Ours)	**42.9**	**81.7**	**62.2**	**68.1**	**61.6**	**67.4**	**66.3**	**59.4**	44.7	**46.7**	**38.0**	**72.4**	**31.6**	**52.3**	**35.4**	**55.0**	**21.8**	**13.6**	**54.9**	**29.4**	**63.4**
50-100m	mAP	CenterPoint‡	20.3	70.9	37.0	48.6	30.0	51.6	12.1	37.2	8.7	11.9	**16.5**	19.0	17.7	17.3	17.4	38.3	16.2	4.5	37.4	0	9.3
		FSD‡	13.2	56.1	33.0	26.7	23.4	30.2	15.2	15.1	11.0	8.2	4.8	15.8	5.7	4.5	12.5	12.5	11.6	5.1	6.2	1.5	16.1
		VoxelNeXt‡	21.6	66.6	34.7	48.5	27.9	57.6	26.8	40.7	15.7	11.8	13.7	21.1	14.3	18.2	16.3	39.7	21.1	7.3	**37.4**	0.3	16.5
		FSTR(Ours)	**27.0**	**70.9**	**41.3**	**54.1**	**36.0**	**66.3**	**34.5**	**46.2**	**19.9**	**18.7**	11.1	**36.6**	**21.7**	**24.5**	**23.0**	**43.8**	**26.7**	**8.9**	32.3	**8.5**	**30.6**
	CDS	CenterPoint‡	15.1	57.9	29.5	33.1	23.3	40.3	8.7	24.7	5.8	8.1	**12.6**	15.5	13.1	11.7	12.1	26.1	12.2	3.1	28.7	0.0	6.9
		FSD‡	9.8	43.8	25.8	19.3	18.2	24.4	11.1	8.9	8.3	6.0	3.2	12.4	4.0	3.1	8.4	8.5	9.2	3.7	4.7	1.2	12.4
		VoxelNeXt‡	15.8	54.0	27.4	33.5	21.3	43.5	21.1	25.4	11.0	8.0	10.1	16.3	10.1	12.5	10.8	26.7	15.2	5.2	**28.1**	0.1	11.7
		FSTR(Ours)	**20.4**	**58.3**	**33.0**	**38.7**	**28.4**	**52.8**	**26.7**	**31.3**	**13.8**	**13.5**	8.3	**29.9**	**16.5**	**16.9**	**16.5**	**30.3**	**20.5**	**6.6**	25.3	**5.7**	**23.0**
100-150m	mAP	CenterPoint‡	10.1	**45.6**	21.0	**26.6**	17.1	17.2	1.2	**31.0**	1.1	**11.6**	7.2	4.0	8.6	4.5	7.6	26.7	4.9	2.8	**19.9**	0	0.6
		FSD‡	3.0	22.8	15.2	3.0	10.6	3.0	0.0	0.0	0.0	1.0	2.2	2.1	4.9	0.2	3.4	1.3	0.1	1.9	0.3	0.0	1.0
		VoxelNeXt‡	9.8	41.6	19.8	23.5	16.5	37.2	2.6	12.3	1.2	7.9	2.4	7.7	8.8	5.3	7.8	25.1	5.0	5.2	15.3	0.2	1.7
		FSTR(Ours)	**12.0**	42.6	**25.0**	25.7	**19.2**	**54.0**	**4.6**	8.4	**2.7**	10.1	**12.2**	**10.5**	**9.2**	**7.7**	**9.7**	**29.2**	**10.5**	**4.9**	10.6	**2.2**	**3.4**
	CDS	CenterPoint‡	7.0	**33.8**	15.6	**17.1**	12.6	12.0	0.7	**17.5**	0.7	**7.4**	5.8	2.9	6.4	2.7	4.8	16.6	3.4	1.8	**15.6**	0.0	0.4
		FSD‡	2.1	16.2	11.3	1.9	7.7	2.0	0.0	0.0	0.0	0.5	1.6	1.6	3.4	0.1	2.2	0.8	0.1	1.3	0.3	0.0	0.6
		VoxelNeXt‡	6.6	30.2	14.5	14.8	11.8	24.1	1.8	6.4	0.7	4.9	1.7	5.4	6.2	3.2	4.7	15.3	3.5	3.4	11.4	0.1	1.1
		FSTR(Ours)	**8.4**	31.7	**19.3**	16.5	**14.0**	**38.8**	**3.0**	3.8	**1.5**	6.7	**10.1**	**7.4**	**6.5**	**4.7**	**6.2**	**18.1**	**7.7**	**3.4**	8.1	**0.9**	**2.3**

TABLE VI

ABLATION STUDY ON THE PROPOSED MODULES: DQS AND ST, GCA, DT, AND GDQ. THE PERFORMANCES ARE EVALUATED ON THE NUSCENES VALIDATION SET. WE TAKE CMT-L [32] AS OUR BASELINE EXPERIMENT

Exp.	Condition						nuScenes Performance							
	Baseline	VoxelNext	DQ. & ST.	GCA.	DT.	GDQ.	mAP	mATE	mASE	mAOE	mAVE	mAAE	NDS	Latency
1.	✓						61.0	0.336	0.257	0.270	0.239	0.188	67.6	72ms
2.	✓	✓					61.6	0.339	0.256	0.279	0.225	0.189	67.9	81ms
3.	✓	✓	✓				62.8	0.317	0.260	0.310	0.249	0.184	68.2	66ms
4.	✓	✓	✓	✓			63.9	0.308	0.267	0.277	0.242	0.188	69.1	67ms
5.	✓	✓	✓	✓	✓		64.3	0.306	0.264	0.292	0.244	0.189	69.2	65ms
6.	✓	✓	✓	✓	✓	✓	**64.4**	0.308	0.257	0.294	0.243	0.185	**69.3**	65ms

the fact that FSTR effectively compensates for the lack of receptive field of the sparse convolutional backbone when the points are too sparse on the far range.

To further validate the advantages of the proposed method for long-range detection, we reproduce the previous works [11], [36], [49] and conduct detailed split-distance evaluation, and the results are shown in Table V.

The results show that the performance of the proposed method on three different range intervals, i.e., 0–50, 50–100, and 100–150 m, is improved by a large margin compared with the existing methods, achieving +8.6%, +5.4%, and +1.9% improvement in mAP, and +6.9%, +4.8%, and +1.4% in CDS metrics, respectively. It can be seen that due to the limitation of the receptive field in sparse convolution backbone, the existing fully sparse methods [36], [49] have a significant disadvantage in large targets compared to methods with dense convolution, such as articulated bus (A-BUS) and normal bus (BUS). In contrast, FSTR efficiently introduces the transformer decoder to bring a larger and more flexible global receptive field, which significantly solves the problem of the insufficient receptive field of fully sparse methods and greatly improves the performance of the detector. It should be noted that the dense prediction detector CenterPoint [11] has a competitive performance in the long-range interval (100–150 m), and we believe it is because the dense convolution alleviates the insufficient receptive field problem in the FSD, but it also costs a lot, which means that it is also not appropriate for the long-range detection task. We will analysis it in detail in Section V.

D. Ablations

We present the ablation studies for the submodule in FSTR in Table VI.

We use CMT-L [32] as the baseline model in Table VI. First, to validate the performance of the proposed method fairly, we first replace the sparse backbone of CMT-L from the popular one with VoxelNeXt. The sparse BEV feature map outputs from VoxelNeXt are densified to a dense feature map to be input to the detection head of CMT-L. Exp. 2 shows that the stronger backbone brings a performance improvement of +0.6 mAP and +0.3 NDS to the detector, and the

TABLE VII

ABLATION STUDIES OF DIFFERENT HYPERPARAMETERS IN THE PROPOSED FSTR. (a) NUMBER OF THE SPARSE DECODER. (b) NUMBER OF THE FOREGROUND TOKEN. (c) RADIUS OF GCA. (d) NUMBER OF THE LEARNABLE QUERIES. (e) NUMBER OF THE DQS

Decoder num	mAP	NDS	Head Latency
1	**60.7**	**66.3**	20ms
2	60.0	66.3	22ms
3	59.7	66.0	24ms
4	58.8	65.9	27ms
5	58.8	66.0	29ms
6	59.4	65.2	31ms

(a)

Tokens	mAP	NDS	Head Latency
32400 (dense)	59.9	65.2	27ms
28000	60.3	66.3	25 ms
14000	60.3	66.2	22 ms
10000	**60.7**	**66.3**	20 ms
7000	60.4	65.8	19 ms
1000	60.2	65.6	17 ms

(b)

Radius w	mAP	NDS
$w(\mathbf{p}) = \delta(\mathbf{p} - \mathbf{r})$	**60.7**	**66.3**
1	60.4	66.1
2	60.1	66.0
3	60.0	66.0
$w(\mathbf{p}) = \delta(\mathbf{p} - p_{nearest,\mathbf{r}_p})$	60.4	65.7

(c)

L. Query	mAP	NDS	H. Latency
0	60.0	65.7	17ms
100	60.5	66.0	18ms
300	**60.7**	**66.3**	20ms
500	60.2	66.1	21ms
700	60.0	65.6	22ms

(d)

D. Query	mAP	NDS	H. Latency
50	60.5	66.2	18ms
100	60.2	66.0	19ms
150	60.5	66.1	19ms
200	**60.7**	**66.3**	20ms
300	60.4	66.0	21ms

(e)

backbone also increases the inference latency by +9 ms. Second, we replace the head of CMT-L with the proposed ST and use both DQ and learnable query as the query of the decoder, which boosts the performance by +1.2 mAP and +0.3 NDS performance gain for the model and reduces the inference of the model by −15 ms in Exp. 3. Next, we add the proposed Gaussian context aggregation module in Exp. 4, which significantly reduces the difficulty in searching and aggregating foreground information in the decoders. This module brings a performance improvement of +1.1 mAP and +0.9 NDS with almost no additional inference latency. After that, we introduce the drop token (DT) operation in ST in Exp. 5 to reduce the influence of the unuseful background information on the query, which has an impact of +0.3 mAP and +0.1 NDS on the model performance and further reduces the inference time by −2 ms. Finally, the GDQ instead of the denoising query is validated in Exp. 6, which yields a performance gain of +0.1 mAP and +0.1 NDS for the model. The denoising queries only appear in the training, which will not bring any inference cost to the detector.

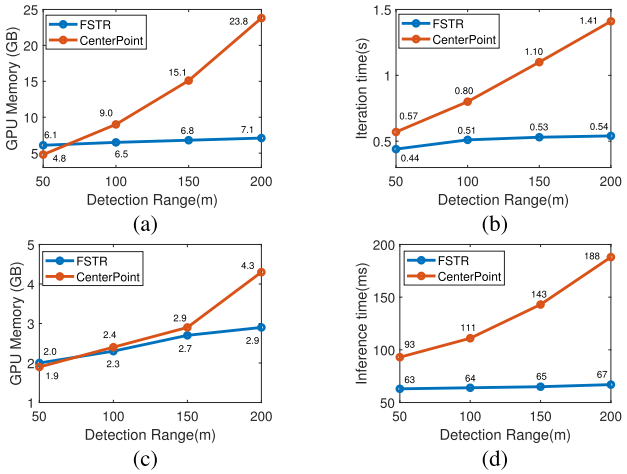

Fig. 5. Latency and memory cost on AV2 and various perception ranges during training and inference. (a) Training memory cost. (b) Training iteration time. (c) Inference memory cost. (d) Inference latency.

There are some hyperparameters in the proposed method, and we further perform ablation experiments on these parameters to explore their effects on the model performance in Table VII. The experiments are conducted for 20 epochs without CBGS [57].

We first explored the effect of the number of decoders on the model performance, as shown in Table VII(a). Increasing the number of decoders leads to a significant decrease in mAP. There is a better performance-efficient trade-off when using only one decoder, which has a +1.3 mAP, +1.1 NDS, and −11-ms inference latency compared with six decoders. We believe this is due to the fact that the DQ already provides good localization and context, which eases the regression of bounding boxes in decoders. Next, we explore the effectiveness of the number of sparse tokens on the detector performance, as shown in Table VII(b). Drop the background token with high confidence properly boost the performance by +0.8 mAP and +0.9 NDS, and reduce the inference latency by −7 ms, while dropping too many tokens will evitably lose information, which results in degradation in the performance. In Table VII(c), the model performance shows a degradation when expanding the radius of the GCA. We believe this is due to the fact that although a large radius expanding the local receptive field, it introduces too much noise as well. There are a improvement boost of +0.7 mAP and +0.6 NDS by optimizing this hyperparameter, as shown in Table VII(c). We also conduct ablation experiments on the effect of the number of both dynamic and learnable queries on the model performance, as shown in Table VII(d) and (e). The learnable queries are two main functions: recall the object missed by the DQs and stable the target assignment of Hungarian Match during training. We set 300 learnable

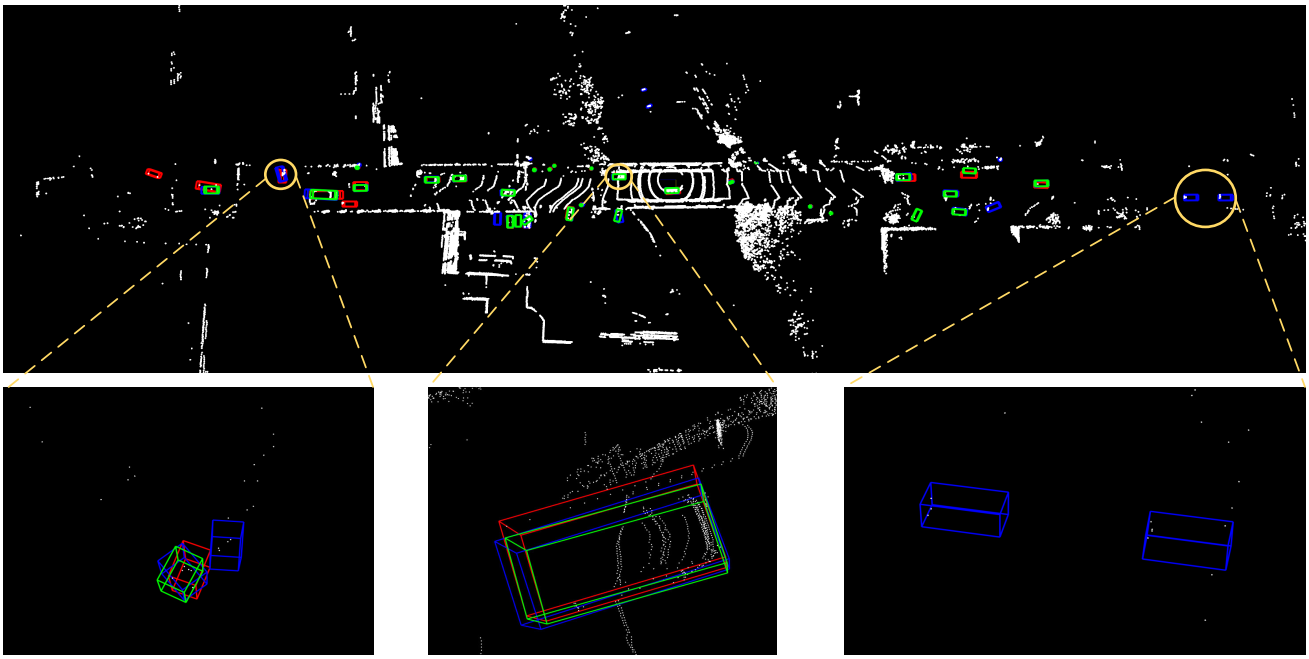

Fig. 6. Visualization of the detection results on the long-range AV2 dataset [21]. Each color represents one kind of 3-D bounding box, red for ground truth, blue for prediction from CenterPoint [11], and green for prediction from FSTR. Three sets of bounding boxes are marked with yellow circles in the figure and enlarged below. FSTR has fewer false positive predictions as well as higher recall compared with the CenterPoint at far distances while having higher regression quality for TPs at both near and far distances.

queries in FSTR, which improves performance by +0.7 mAP and +0.6 NDS. On the other head, the DQs also should recall the interested object as much as possible. The experiments show that 200 DQs are enough for the proposed method, which boosts the performance by +0.2 mAP and +0.1 NDS.

V. Analysis

We further analyze the method on the training cost, inference cost, and prediction visualization to further validate the advantages of full sparse architecture for long-range detection in FSTR. This part of the experiment is conducted based on the AV2 dataset. CenterPoint, the most popular detector on the 3-D detection task, is used as the baseline in this section.

Fig. 5(a) and (b) shows the memory costs and iteration time curves of FSTR and CenterPoint with different detection ranges during training. FSTR only cost 30% GPU memory and 38% iteration time during training compared with the baseline method. Similarly, Fig. 5(c) and (d) shows the memory cost and inference latency of the two methods during the inferencing. FSTR costs 67% GPU memory and 36% latency during inference compared with the baseline method and achieves real-time perceptron in the long-range detection task. Since the size of the BEV feature map grows squarely with the increase of the detection range, the efficiency of the dense prediction detector CenterPoint decreases drastically both during the training and inference. In contrast, benefiting from the fully sparse framework of the FSTR and the decoupling between DQ number and detection range, the long-range detection range only brings a little impact on the efficiency of FSTR during training and inference, and the excellent performance of the proposed method on long-distance detection is also demonstrated in Table IV.

To further validate the effectiveness of the proposed method, we further perform a qualitative visualization of the prediction results of FSTR on the long-range detection task and compare them with the baseline method, as shown in Fig. 6. Each color represents a 3-D bounding box from a different source, where red represents ground truth, blue represents the prediction from CenterPoint, and green represents the prediction from FSTR. To the convenience of observation, we zoom in on the three sets of prediction bounding boxes and displayed them at the bottom of Fig. 6. It can be seen that for objects in the close range, FSTR has better regression quality than the baseline method. For the objects in the middle range and long range, FSTR has better regression quality, better recall, and outputs fewer false positive predictions compared with the baseline method.

VI. Conclusion

In this article, we present a fully sparse LiDAR-based 3-D object detection framework. It is flexible and compatible with the existing sparse 3-D backbone and directly uses sparse voxel features as input tokens to the decoder to get rid of the dependence on dense BEV feature map. We propose to use DQ to provide certain a priori positional and contextual information to the transformer decoder in order to reduce the search difficulty of the decoder in inference, which, in turn, makes it possible to achieve better detection performance using a smaller network size. Our FSD achieves competitive performance as well as real-time inference on the popular nuScenes dataset and achieves far better performance than current methods on the large-scale long-distance dataset AV2, realizing a good efficiency–accuracy trade-off. We hope such a simple pipeline design could provide a strong baseline for the FSD and more insights into 3-D object detection.

REFERENCES

[1] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *J. Field Robot.*, vol. 37, no. 3, pp. 362–386, Apr. 2020.

[2] D. J. Yeong, G. Velasco-Hernandez, J. Barry, and J. Walsh, "Sensor and sensor fusion technology in autonomous vehicles: A review," *Sensors*, vol. 21, no. 6, p. 2140, Mar. 2021.

[3] Y. Zhou and O. Tuzel, "VoxelNet: End-to-end learning for point cloud based 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4490–4499.

[4] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, p. 3337, Oct. 2018.

[5] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, and H. Li, "Voxel R-CNN: Towards high performance voxel-based 3D object detection," 2020, *arXiv:2012.15712*.

[6] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast encoders for object detection from point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 12689–12697.

[7] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2999–3007.

[8] H. Wu, J. Deng, C. Wen, X. Li, C. Wang, and J. Li, "CasA: A cascade attention network for 3-D object detection from LiDAR point clouds," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5704511.

[9] D. Zhang, X. Wang, Z. Zheng, X. Liu, and G. Fang, "ARFA: Adaptive reception field aggregation for 3D detection from LiDAR point cloud," *IEEE Sensors J.*, vol. 23, no. 11, pp. 11156–11167, Jun. 2023.

[10] D. Zhang, X. Wang, Z. Zheng, and X. Liu, "Unsupervised domain adaptive 3-D detection with data adaption from LiDAR point cloud," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5705814.

[11] T. Yin, X. Zhou, and P. Krähenbühl, "Center-based 3D object detection and tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 11779–11788.

[12] L. Fan et al., "Embracing single stride 3D object detector with sparse transformer," 2021, *arXiv:2112.06375*.

[13] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 77–85.

[14] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," 2017, *arXiv:1706.02413*.

[15] S. Shi, Z. Wang, J. Shi, X. Wang, and H. Li, "From points to parts: 3D object detection from point cloud with part-aware and part-aggregation network," 2019, *arXiv:1907.03670*.

[16] S. Shi, X. Wang, and H. Li, "PointRCNN: 3D object proposal generation and detection from point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 770–779.

[17] Z. Li, F. Wang, and N. Wang, "LiDAR R-CNN: An efficient and universal 3D object detector," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 7542–7551.

[18] Y. Zhang, Q. Hu, G. Xu, Y. Ma, J. Wan, and Y. Guo, "Not all points are equal: Learning highly efficient point-based detectors for 3D LiDAR point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 18931–18940.

[19] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 91–99.

[20] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "CenterNet: Keypoint triplets for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6568–6577.

[21] B. Wilson et al., "Argoverse 2: Next generation datasets for self-driving perception and forecasting," 2023, *arXiv:2301.00493*.

[22] H. Caesar et al., "NuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11618–11628.

[23] B. Liu, M. Wang, H. Foroosh, M. Tappen, and M. Penksy, "Sparse convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 806–814.

[24] B. Graham and L. van der Maaten, "Submanifold sparse convolutional networks," 2017, *arXiv:1706.01307*.

[25] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[26] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 213–229.

[27] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable DETR: Deformable transformers for end-to-end object detection," 2020, *arXiv:2010.04159*.

[28] F. Li, H. Zhang, S. Liu, J. Guo, L. M. Ni, and L. Zhang, "DN-DETR: Accelerate DETR training by introducing query denoising," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 13609–13617.

[29] D. Meng et al., "Conditional DETR for fast training convergence," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 3631–3640.

[30] B. Zhu, Z. Wang, S. Shi, H. Xu, L. Hong, and H. Li, "ConQueR: Query contrast voxel-DETR for 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 9296–9305.

[31] Y. Liu, T. Wang, X. Zhang, and J. Sun, "PETR: Position embedding transformation for multi-view 3D object detection," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2022, pp. 531–548.

[32] J. Yan et al., "Cross modal transformer: Towards fast and robust 3D object detection," 2023, *arXiv:2301.01283*.

[33] X. Bai et al., "TransFusion: Robust LiDAR-camera fusion for 3D object detection with transformers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 1080–1089.

[34] S. Shi et al., "PV-RCNN++: Point-voxel feature set abstraction with local vector representation for 3D object detection," 2021, *arXiv:2102.00463*.

[35] G. Shi, R. Li, and C. Ma, "PillarNet: Real-time and high-performance pillar-based 3D object detection," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2022, pp. 35–52.

[36] Y. Chen, J. Liu, X. Zhang, X. Qi, and J. Jia, "VoxelNeXt: Fully sparse VoxelNet for 3D object detection and tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 21674–21683.

[37] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, pp. 1–12, Oct. 2019.

[38] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point transformer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 16239–16248.

[39] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, "PCT: Point cloud transformer," *Comput. Vis. Media*, vol. 7, no. 2, pp. 187–199, Jun. 2021.

[40] W. Shi and R. Rajkumar, "Point-GNN: Graph neural network for 3D object detection in a point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1708–1716.

[41] C. R. Qi, O. Litany, K. He, and L. Guibas, "Deep Hough voting for 3D object detection in point clouds," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9276–9285.

[42] Z. Yang, Y. Sun, S. Liu, and J. Jia, "3DSSD: Point-based 3D single stage object detector," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11037–11045.

[43] Q. Hu et al., "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11105–11114.

[44] S. Shi, Z. Wang, J. Shi, X. Wang, and H. Li, "From points to parts: 3D object detection from point cloud with part-aware and part-aggregation network," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 8, pp. 2647–2664, Aug. 2021.

[45] S. Shi et al., "PV-RCNN: Point-voxel feature set abstraction for 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10526–10535.

[46] W. Liu et al., "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 21–37.

[47] H. Law and J. Deng, "CornerNet: Detecting objects as paired keypoints," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 734–750.

[48] Y. Chen, Y. Li, X. Zhang, J. Sun, and J. Jia, "Focal sparse convolutional networks for 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 5418–5427.

[49] L. Fan, F. Wang, N. Wang, and Z.-X. Zhang, "Fully sparse 3D object detection," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 351–363.

[50] H. Zhang et al., "DINO: DETR with improved DeNoising anchor boxes for end-to-end object detection," 2022, *arXiv:2203.03605*.

[51] Q. Chen et al., "Group DETR: Fast DETR training with group-wise one-to-many assignment," 2022, *arXiv:2207.13085*.
[52] F. Chen, H. Zhang, K. Hu, Y.-K. Huang, C. Zhu, and M. Savvides, "Enhanced training of query-based object detection via selective query recollection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 23756–23765.
[53] J. S. K. Hu, T. Kuai, and S. L. Waslander, "Point density-aware voxels for LiDAR 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 8459–8468.
[54] Z. Zhou, X. Zhao, Y. Wang, P. Wang, and H. Foroosh, "CenterFormer: Center-based transformer for 3D object detection," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2022, pp. 496–513.
[55] Z. Liu et al., "Swin Transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 9992–10002.
[56] S. Wang, X. Jiang, and Y. Li, "Focal-PETR: Embracing foreground for efficient multi-camera 3D object detection," 2022, *arXiv:2212.05505*.
[57] B. Zhu, Z. Jiang, X. Zhou, Z. Li, and G. Yu, "Class-balanced grouping and sampling for point cloud 3D object detection," 2019, *arXiv:1908.09492*.
[58] Q. Chen, L. Sun, Z. Wang, K. Jia, and A. Yuille, "Object as hotspots: An anchor-free 3D object detection approach via firing of hotspots," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 68–84.
[59] Y. Hu et al., "AfDetV2: Rethinking the necessity of the second stage for object detection from point clouds," in *Proc. AAAI Conf. Artif. Intell.*, 2022, vol. 36, no. 1, pp. 969–979.
[60] S. Deng, Z. Liang, L. Sun, and K. Jia, "VISTA: Boosting 3D object detection via dual cross-VIew SpaTial Attention," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2022, pp. 8448–8457.
[61] Y. Li, Y. Chen, X. Qi, Z. Li, J. Sun, and J. Jia, "Unifying voxel-based representation with transformer for 3D object detection," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 18442–18455.
[62] Y. Chen, J. Liu, X. Zhang, X. Qi, and J. Jia, "LargeKernel3D: Scaling up kernels in 3D sparse CNNs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 13488–13498.
[63] Y. Li et al., "Fully sparse fusion for 3D object detection," 2023, *arXiv:2304.12310*.
[64] T. Dao, D. Fu, S. Ermon, A. Rudra, and C. Ré, "FlashAttention: Fast and memory-efficient exact attention with IO-awareness," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 16344–16359.
[65] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–12.
[66] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

Diankun Zhang received the B.E. degree from the Beijing University of Technology, Beijing, China, in 2019. He is currently pursuing the Ph.D. degree with the Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing.

His research interests include 3-D computer vision and autonomous driving perception.

Zhijie Zheng received the B.E. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2019. He is currently pursuing the Ph.D. degree with the University of Chinese Academy of Sciences, Beijing, China, and the Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing.

His research interests include perception beyond the visible spectrum, deep learning, and cross-modal learning.

Haoyu Niu received the B.E. degree from Shandong University, Jinan, China, in 2020. He is currently pursuing the M.D. degree in computer science with the University of Chinese Academy of Sciences, Beijing, China.

His research interests include deep-learning-based 3-D object detection and tracking.

Xueqing Wang received the B.Eng. degree from the University of Science and Technology Beijing, Beijing, China, in 2020. She is currently pursuing the M.D. degree with the Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing.

Her research interests include deep-learning-based signal processing and 3-D object detection.

Xiaojun Liu (Member, IEEE) received the B.S. and M.S. degrees in electronics science and technology from the North University of China, Taiyuan, China, in 1995 and 1998, respectively, and the Ph.D. degree in electrical engineering from the Chinese Academy of Sciences (CAS), Beijing, China, in 2001.

Since 2001, he has been with the Key Laboratory of Electromagnetic Radiation and Sensing Technology, CAS, where he is currently a Professor and leads a group on the studies of remote sensing of ice sheets with projects supported by the National Natural Science Foundation of China and the National High Technology Research and Development Projects (863 Projects) of China. He developed several radar systems currently being used at the Institute of Electronics, Chinese Academy of Sciences (IECAS), Beijing, for sounding and imaging of polar ice sheets. These radar systems have been used in field experiments in Antarctica for five times since CHINARE 26. In the field of lunar and deep space exploration, he leads a group to demonstrate the scheme of subsurface penetrating radar of Mars Orbiter and Vehicle for China's first Mars exploration. The program has been adopted and is being implemented. His research interests include application of radars to the remote sensing of the polar ice sheets, ionosphere, and land.