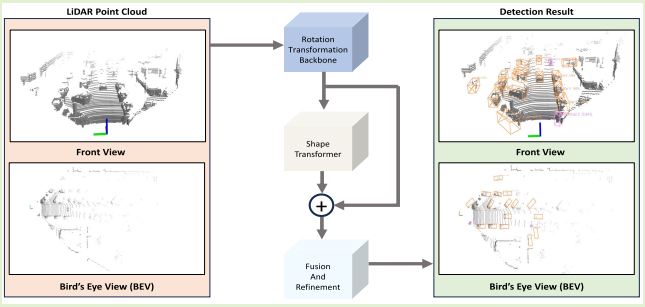
TSSTDet: Transformation-Based 3-D Object Detection via a Spatial Shape Transformer

Hiep Anh Hoang, Duy Cuong Bui, and Myungsik Yoo

***Abstract*—Accurately detecting and understanding the shapes of objects in 3-D scenes are essential for autonomous driving. In a 3-D scene, objects are distributed with various incomplete shapes and rotations. Determining the shape allows for a comprehensive understanding of an object's dimensions, rotations, and spatial relationships with its surroundings. Traditional detection methods do not explicitly consider the rotations and complete shapes that objects can assume. Consequently, these methods require large networks and extensive data augmentation to detect accurately. Taking advantage of the vision-transformer (ViT), we introduce an efficient transformer-based 3-D detector called transformation-based 3-D object detection via a spatial shape transformer (TSSTDet) to address these challenges. We constructed TSSTDet as a multistage detector based on a light detection and ranging (LiDAR) point cloud. Specifically, TSSTDet utilizes a sparse convolution (SpConv) backbone to extract multichannel and transformation-equivariant voxel features. Furthermore, we designed an efficient module that employs the transformer approach to estimate the completed shape of an object. These features are then aligned and aggregated to create lightweight and compact representations that enable high-performance 3-D object detection. We assessed the effectiveness of the proposed framework by evaluating its performance on both the KITTI and Waymo open datasets (WODs). These evaluations demonstrated that our framework achieves top-tier performance in 3-D object detection.**



***Index Terms*—3-D object detection, autonomous driving, light detection and ranging (LiDAR) point cloud, vision transformer.**

I. Introduction

RESEARCH on 3-D object detection in autonomous vehicles from light detection and ranging (LiDAR) data is highly important, as it enables safe and efficient navigation. Unlike 2-D object detection, 3-D object detection expands the search space from a 2-D image plane to a 3-D space. A key aspect of 3-D object detection in self-driving is identifying the category of interest (such as car, pedestrian, and cyclist) by a classification task and localizing objects in a scene by regressing their 3-D spatial bounding box. Precisely detecting and recognizing objects in this 3-D space is necessary for a self-driving car to be aware of the environment and to make correct judgments in smart cities and urban areas.

Manuscript received 24 December 2023; revised 4 January 2024; accepted 4 January 2024. Date of publication 12 January 2024; date of current version 29 February 2024. This work was supported by the National Research Foundation of Korea (NRF) through the Government of South Korea (MSIT) under Grant NRF-2021R1A2B5B01002559. The associate editor coordinating the review of this article and approving it for publication was Dr. Guofa Li. *(Corresponding author: Myungsik Yoo.)*

Hiep Anh Hoang and Duy Cuong Bui are with the Department of Information Communication Convergence Technology, Soongsil University, Seoul 06978, South Korea (e-mail: hiepbk97@soongsil.ac.kr; cuonga1@soongsil.ac.kr).

Myungsik Yoo is with the School of Electronic Engineering, Soongsil University, Seoul 06978, South Korea (e-mail: myoo@ssu.ac.kr).

Digital Object Identifier 10.1109/JSEN.2024.3350770

LiDAR point clouds are largely used for ambient perception in current 3-D object detection approaches. A precise 3-D map of the environment is produced using this point cloud, which provides precise knowledge about objects and their locations from the sensor. Utilizing this 3-D point cloud map as input, cutting-edge 3-D detection methods can effectively identify objects of interest, thereby enhancing perception within the 3-D environment.

Current approaches in this field can be categorized into single-stage or multistage 3-D detection methodologies. Single-stage methods employ encoded features obtained directly from point clouds for object detection [1], [2], [3], [4]. In contrast, multistage approaches construct their models using a region-based convolutional neural network (RNN) framework, as in [5]. This method involves generating a set of potential bounding boxes, followed by the classification and refinement of each candidate box. Several recent studies have favored the multistage framework, owing to its superior accuracy [6], [7], [8], [9], [10]. For instance, Voxel-RCNN [7] and Point RCNN [10] produce and enhance region proposals by utilizing voxel and point features, respectively.

Nonetheless, the widespread adoption of multistage 3-D detectors utilizing LiDAR technology has encountered significant challenges. The first challenge arises from missing object shapes resulting from occlusions or sensor-signal disruptions.

The second challenge pertains to the diversity of object orientations. These factors collectively contribute to the loss of essential geometric and semantic details, thereby impeding precise object recognition.

To address the first challenge, point cloud prediction and shape-estimation methods, such as SIENet [11], estimate the complete shapes of the foreground within proposed scenes to obtain structural information. This enables the acquisition of representative features for subsequent box refinement. To some extent, BtcDet [12] addresses the challenges of incomplete point cloud structures. These studies demonstrate that enhancing the completeness of a point cloud structure can enhance the precision of 3-D object detection. Nonetheless, existing methods for reconstructing spatial structures inevitably lead to higher computational demands and reasoning times for the detector, especially when employing the conventional convolutional neural network (CNN) paradigm with deeper network architectures. The advantages offered by transformers with multihead attention in the field of computer vision [13] have ushered in numerous promising advancements and have paved the way for addressing this challenge in a novel manner.

The second challenge relates to the numerous possible rotations that objects can take within a 3-D space. In 3-D scenes, objects display a vast spectrum of orientations. That is, if an object alters its orientation within the input data points, its detected bounding box should retain its shape while aligning the angle to match. A 3-D detector's predictions should be equivariant concerning rotations and transformation reflections. Unfortunately, traditional detectors do not explicitly address the variety of rotation and reflection transformations, which can potentially result in unreliable prediction outcomes when addressing changed point clouds.

Recently, some detectors have achieved approximate transformation equivariance through data augmentation [8], [10]. However, the production of huge training samples and the use of more sophisticated networks with larger capacities substantially influence their effectiveness. Recently, equivariant neural networks [14], [15] that explicitly model the transformation equivariance have been developed. The equivariant design has produced optimistic results in autonomous driving. To accomplish this, they converted the input data using various rotation bins and represented equivariance using shared convolutional networks. TED [6], a recent multiple-based method, has demonstrated encouraging outcomes when handling transformation equivariance and invariance in 3-D object detection. However, TED cannot fully address the challenges associated with occlusion.

In this study, we introduce a novel 3-D detection network known as transformation-based 3-D object detection via a spatial shape transformer (TSSTDet) to enhance the performance of 3-D object detection models. Our research primarily focuses on two major issues that previous 3-D object detectors have faced. Specifically, TSSTDet is a multistage framework that can handle the orientation and missing points of the shape problems of an object. TSSTDet consists of a 3-D rotational-transformation backbone tasked with extracting diverse rotation-equivariant features to address object-rotation challenges. In addition, we designed a transformer-based deep network, called a voxel-point shape transformer (VPST), to reconstruct object shapes, primarily to address the occlusion problems in autonomous driving scenarios. Furthermore, TSSTDet integrates an aided network called the attention-fusion and refinement (AFR) module to aggregate features from the preceding steps, thereby enhancing the object-proposal confidence.

Our contributions can be summarized as follows.

1) We present the TSSTDet framework, a robust multistage approach for an efficient rotational-transformation 3-D object detector for object-geometry modeling.
2) We propose an efficient transformer-based module called the VPST, which is capable of dealing with occlusion challenges by reconstructing the complete shape of an object.
3) Our model surpasses the performance of current cutting-edge models across varying levels of complexity within the car, pedestrian, and cyclist categories. Our achievements on the KITTI leaderboard for 3-D object detection are also notably impressive.

II. Related Work

A. Mainstream 3-D Object Detection

The 3-D object detection methods can be categorized, based on their data-processing approaches, into LiDAR-based methods and multimodal-based methods.

1) LiDAR-Based 3-D Object Detection: Significant research has been conducted on 3-D object detection using LiDAR technology over the past few years. Recently, two primary approaches, voxel-based [1], [2], [7], [16] and point-based set abstraction [10], [17], [18], have emerged as key methods for crafting efficient detection frameworks.

Voxel-based approaches represent a scene using voxels. Voxels divide the 3-D space into regular grid cells and encode information about occupancy or object attributes within each cell. The pioneering VoxelNet [2] utilized a voxel-based representation and a 3-D CNN to detect objects in point clouds. It operated on a fixed-size voxel grid and performed 3-D convolutions to extract features and predict object attributes. SECOND [1] employs sparse convolutional (SpConv) layers to handle sparse and irregular point cloud representations. To overcome the limitations of 3-D CNN layers, PointPillars [19] transforms voxels into pillars that are arranged from a bird's-eye view (BEV) perspective, utilizing pseudoimages as the representation.

Point-based approaches operate directly on individual points in a point cloud without voxelization. Point-based methods have been applied to both multistage [8], [10], [20] and single-stage [3], [21] methods. PointNet [17] is a ground-breaking deep-learning architecture capable of detecting objects in unordered point clouds. By employing symmetric functions and a shared multilayer perceptron (MLP) network, PointNet effectively extracts features from individual points. This method achieved a notable performance in 3-D segmentation and object-classification tasks. PointRCNN [10] incorporates a region proposal network (RPN) and a second-stage network that performs region-of-interest (RoI) pooling and point-wise

feature learning. By exploiting both local and global contextual information, PointRCNN improves the accuracy of object detection tasks.

2) Multimodal-Based 3-D Object Detection: Multimodal-based 3-D object detection involves integrating both 2-D and 3-D data. These multimodal methods harness the synergies between 2-D image-based and 3-D point cloud-based detectors. In the initial stages of this approach, techniques emerged that extended features derived from LiDAR points with image-based features to enhance 3-D object detection. For instance, MV3D [22] integrates 2-D and 3-D object detection by fusing information from images and LiDAR data. F-PointNet [23] extends PointNet to incorporate 2-D image features, thereby improving the detection accuracy.

Certain studies [24], [25] independently encoded features from two modalities and then merged these features within a local RoI or BEV plane. Typically, aggregated-view object detection (AVOD) [26] utilizes feature extractors to obtain features from both BEV feature maps and 2-D RGB images before aggregating multimodal features to produce 3-D object proposals. Recently, some studies [27], [28] fused camera images and LiDAR point clouds via virtual points to exploit the advantages of a depth-estimation task. However, virtual points are highly concentrated and frequently contain noise, degrading the performance of the 3-D detector.

B. 3-D-Object Shape Reconstruction

Point cloud reconstruction and completion are essential tasks in computer vision and 3-D scene understanding with the aim of recovering missing or incomplete parts of a point cloud representation. Owing to occlusions, light reflections, and restrictions on viewing angles and resolution, raw point clouds produced by LiDAR are sparse and lacking in geometric and semantic information. Hence, the missing pieces of 3-D shapes should be generated from a partially observed point cloud or with variable levels of noise for the 3-D object detector.

Recent research has focused on deep-learning-based methodologies that aim to determine the entire point cloud using the extracted features and the network's capacity for representational modeling. A two-stage generation procedure with a folding architecture was introduced by FoldingNet [29], with the presumption that 3-D objects can be recovered from 2-D manifolds. Subsequently, SA-Net [30] introduced a hierarchical folding approach in a multistage point-generation decoder. Nevertheless, this presents challenges in terms of interpreting and constraining the implicit representation of the complete shape across intermediate layers, which hinders shape refinement within the local region.

In addition, to randomly group points in point cloud completion tasks, TopNet [31] uses a decoder in a hierarchical rooted-tree form. The implicit intermediary in decoding SA-Net and TopNet, similar to FoldingNet, is a point feature that denotes the form structure, which is challenging to constrain explicitly. To overcome this issue, SnowflakeNet [32] utilizes snowflake point deconvolution (SPD) to extract parent patterns and feed them across to child points via a pointwise-splitting procedure. This enables the network to produce a precise geometry of the objects.

With the rapid emergence of transformer-based methods, ShapeFormer [33] introduces a vector-quantized deep implicit function (VQDIF) to acquire a sparse representation of partial point clouds and employs transformer-based techniques to generate complete shapes. Moreover, AutoSDF [34] presents an autoregressive generation method over a discrete distribution to complete the object shape from a partial point cloud. Thanks to recent advancements in 3-D object reconstruction, 3-D detectors can incorporate the capability to infer missing points within obscured shapes.

C. Transformer-Based Object Detection

Inspired by the remarkable success of transformer architectures in natural language processing (NLP), researchers have recently extended their applications to computer-vision (CV) tasks. Although CNNs have long been regarded as foundational components in vision applications [35], [36], transformers are now emerging as a promising alternative. Image-generative pretrained transformer (Image GPT) [37] pioneered the introduction of transformers to 2-D image-classification tasks through unsupervised learning. Furthermore, the vision-transformer (ViT) [13] represents a pure transformer approach that is directly applied to sequences of image patches for image-classification tasks. For high-level vision tasks, certain object detection methods have delved into the potential of self-attention mechanisms and have proceeded to improve the pertinent modules for modern detectors, such as the feature-fusion module [38] and the prediction head [39].

In 3-D vision, a point-attention transformer (PAT) [40] introduced Gumbel subset sampling and group-shuffle attention for permutation-invariant tasks within point clouds. CT3D [41] utilizes a high-quality region-proposal network (RPN) and channel-wise transformer architecture for improved accuracy and minimal handcrafted design. Voxel transformer (VoTr) [16] introduces a voxel-based transformer backbone to address the issue of limited receptive fields, allowing self-attention to establish long-range relationships between voxels. A cascade attention network (CasA) [39] introduced a method that gradually improved and supplemented predictions by utilizing multiple subnetworks to achieve high-quality predictions. CasA improves the accuracy of proposal refinement by considering the quality of proposals from all preceding stages, while consolidating object features across several phases. These studies served as inspiration for the approach used in this study, which merges features gathered from different subnetworks to produce better 3-D object detection.

III. Methodology

This section provides a comprehensive description of the proposed model. In Section III-A, we provide an overview of the proposed architecture. Section III-B outlines the elements of Stage 1, encompassing the rotational-transformation backbone, rotation-transformation pooling, and RPN. Section III-C describes the core structure of Stage 2, which enhances the geometric attributes of objects. Section III-D describes the AFR network responsible for synthesizing features and refining the final prediction of a 3-D object detection task.

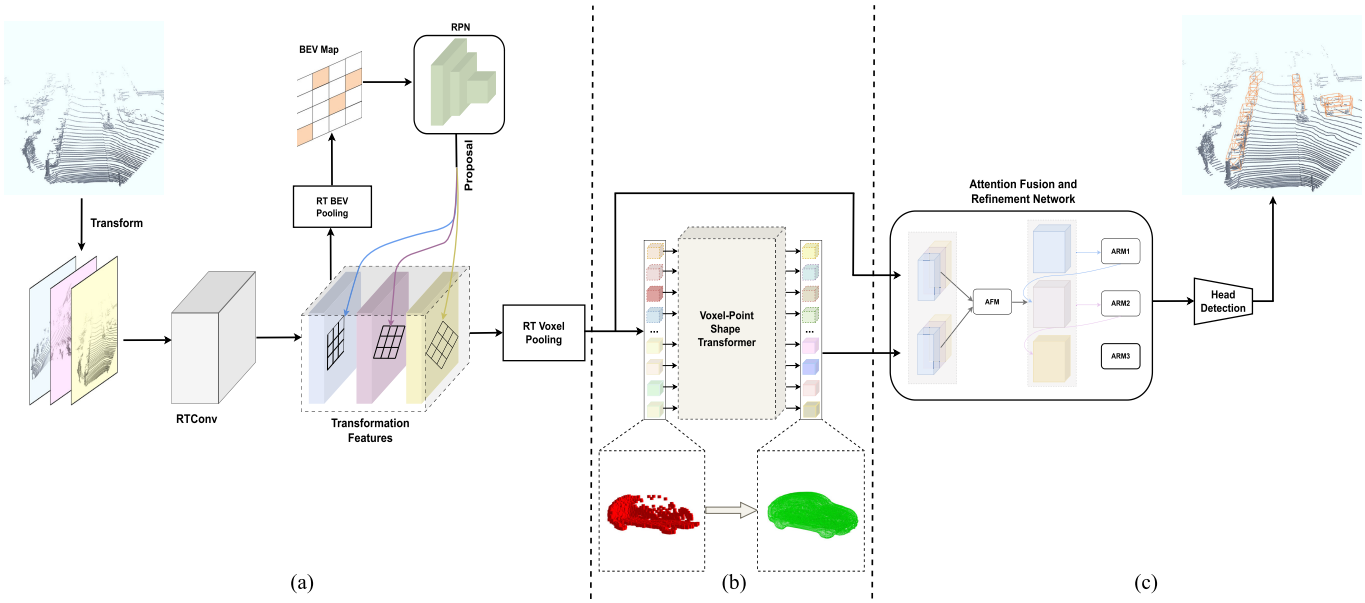


Fig. 1. Our TSSTDet multistage-framework architecture. (a) 3-D RTConv backbone, RT BEV pooling, and RT voxel pooling are applied on multiple rotated point clouds to capture the multi-channel rotational-transformation features. (b) VPST reconstructs the completed shape from a partial observation. (c) Attention fusion and refinement network aggregates multiple features for proposal refinement.

Section III-E describes the loss functions employed in the proposed model.

A. Overall Architecture

The overall architecture of the proposed method is illustrated in Fig. 1. Our aim is to develop a robust 3-D detector for outdoor scenarios in which objects of interest may be obscured and exhibit varying orientations. To address this challenge, we introduced a 3-D sparse rotation-transformation convolution backbone to effectively capture orientation features. Furthermore, we propose a VPST module to address the issue of missing object shapes.

Our approach involves adopting a resilient transformation-equivariant method, empowering the model to gain insights into the object's orientation, while also gaining insight into the object's shape through shape reconstruction. This understanding enables the model to accurately determine the necessary location for synthesizing the object's pattern.

To achieve this, we adopted a multistage approach. In the first stage, a 3-D rotation-transformation convolution backbone encodes a LiDAR point cloud that yields transformation-equivariant features. These features are then pooled by the transformation BEV pooling to generate BEV feature maps. Subsequently, an RPN is employed to generate proposal bounding boxes for the objects. We then utilize transformation voxel pooling to incorporate transformation-invariant features into the proposals for the next stage. In the next stage, we present a VPST (shown in Fig. 2) to address the challenge of missing shape information for foreground objects. Specifically, we transform the voxel coordinates of foreground objects from a 3-D space into an $n \times n \times n$ grid space. Subsequently, a transformer network is used to estimate the complete shape of the object. Finally, we designed an attention-fusion module (AFM) to consolidate the feature map, and a multi-attention refinement module to fine-tune the final precise prediction.

B. Stage 1: Rotational-Transformation Feature Extraction

1) Rotational-Transformation Backbone: Most 3-D detectors employ point- or voxel-based methods; however, these conventional approaches lack rotation and translation equivariance. We introduce a rotational-transformation convolution (RTconv) backbone to efficiently encode raw points while ensuring transformation equivariance. RTconv was constructed based on the widely adopted SpConv [42]. Although SpConv exhibits translation equivariance similar to that of CNNs, it lacks equivariance to rotation. To overcome this issue, we incorporated additional rotational channels that enabled the adaptation of rotation-equivariant features. Our RTConv facilitates the learning of object-level equivariant features in outdoor scenes.

With the input point cloud denoted as $\mathcal{P}$, the rotational-transformation backbone assists the detector D^{ϕ} using the rotational-transformation action $\mathcal{T}$ in detecting the bounding box B as follows:

$$D^{\phi}[\mathcal{T}(\mathcal{P})] = \mathcal{T}[D^{\phi}(\mathcal{P})], \quad \mathcal{T} \in G$$
$$G \cong \oplus \rtimes \omega \tag{1}$$

where G represents a rotational-transformation group comprising the 2-D BEV translation set ($\oplus \in \mathbb{R}^2$) and 2-D BEV rotation-reflection set ω. In particular, ω encompasses a reflection action set $\{(\pm 1, *)\}$ and a discrete rotation set $\{\phi_i, i \in (0, 2\pi)\}$. Considering N discrete rotation cases, each multiplied by the reflection case, ω forms a discrete subgroup of order $2N$.

According to [15], we applied the transformation actions $\{\mathcal{T}_k\}_{k=1}^{2N}$ to transform $\mathcal{P}$, resulting in the generation of $2N$ sets, denoted as $\{\mathcal{P}^{\mathcal{T}_k}\}_{k=1}^{2N}$. A mean voxel feature-extraction process is then applied to all points, producing mean voxel sets $\{\hat{\mathcal{P}}^{\mathcal{T}_k}\}_{k=1}^{2N}$. Subsequently, we utilized SpConv $\Psi(\cdot)$ with a filter configuration of (16, 32, 64, 64) to encode the mean

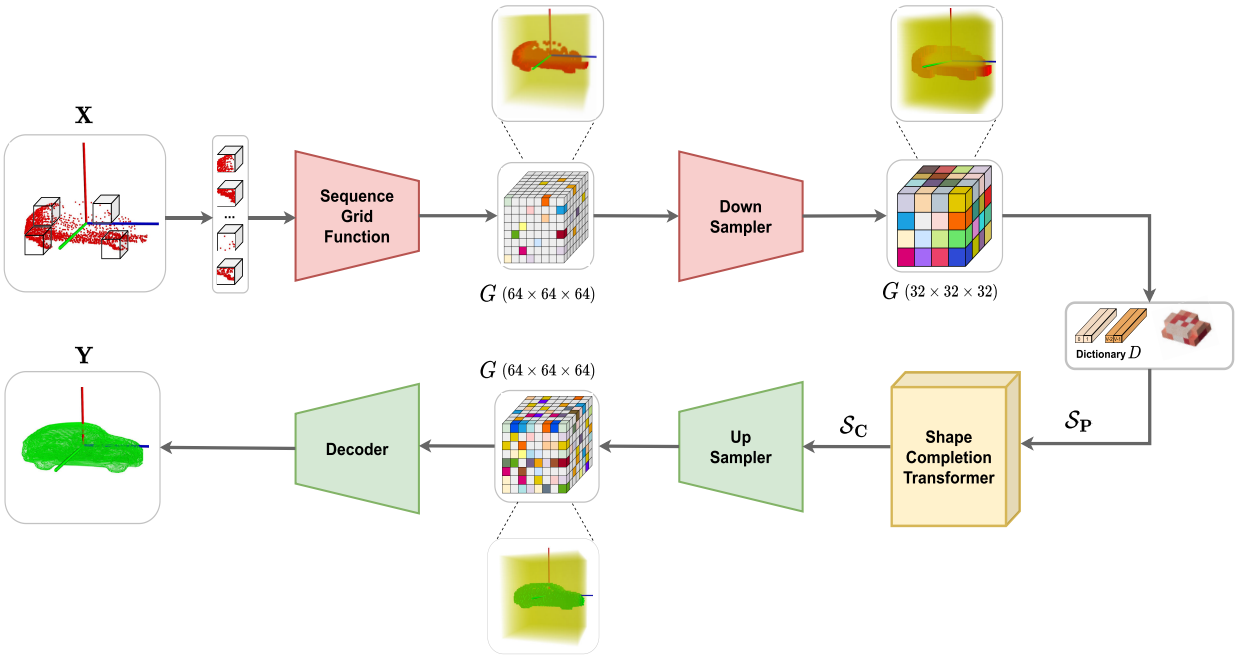


Fig. 2. Overview of our VPST. Given a partial voxel shape *V*, our VPST encoder first uses a patch-wise on the object shape to convert it to a grid feature sequence. Subsequently, a downsampler reduces the grid's dimension. These features are then substituted with the indices of their nearest neighbors in a learned dictionary *D*, forming a compact-sequence discrete tuple pair comprising the coordinate and quantized-feature index value. An autoregressive transformer yields a complete sequence for the object shape. Finally, the upsampler and decoder transform the sequence into point cloud features.

voxels into rotational-transformation voxel (RT voxel) features

$$\mathcal{V}^{\mathcal{T}_k} = \Psi\left(\hat{P}^{\mathcal{T}_k}\right), \quad k = 1, 2, \ldots, 2N. \tag{2}$$

In contrast to the voxel features extracted by normal SpConv, the features $\{\mathcal{V}^{\mathcal{T}_k}\}_{k=1}^{2N}$ contain a variety of properties related to rotation and symmetric translation.

2) Rotational-Transformation Pooling: Pooling operations play a vital role in the multistage approach by reducing the spatial resolution and expediting computation. However, regular pooling methods may not effectively adapt to features derived from RTConv. To address this issue, we propose rotational-transformation BEV (RT BEV) and RT voxel pooling.

First, RT BEV pooling was designed to align and consolidate scene-level voxel features into a concise BEV map, through a combination of bilinear interpolation and max-pooling. In essence, it involves compressing the voxel features denoted as $\{\mathcal{V}^{\mathcal{T}_k}\}_{k=1}^{2N}$ into BEV features $\{\mathcal{B}^{\mathcal{T}_k}\}_{k=1}^{2N}$ along the height dimension. The BEV features must be aligned to the same coordinate system because they have been acquired through various transformations. To achieve this, we convert the grid points into a BEV coordinate system, resulting in the creation of a new set of grid points, denoted as $\{X^{\mathcal{T}_k}\}_{k=1}^{2N}$. This transformation process is performed in accordance with the set of transformation actions $\{\mathcal{T}_k\}_{k=1}^{2N}$.

To ensure a precise spatial alignment and maintain consistency in the transformation process, we apply a series of bilinear interpolations Γ to the BEV coordinates to obtain a set of aligned features $\{\mathcal{F}_a^{\mathcal{T}_k}\}_{k=1}^{2N}$. The interpolated feature is padded by zeroes if the border pixel in $\mathcal{B}^{\mathcal{T}_1}$ has no matching pixel in $\mathcal{B}^{\mathcal{T}_2}, \ldots, \mathcal{B}^{\mathcal{T}_N}$. The aligned features were calculated as follows:

$$\mathcal{F}_a^{\mathcal{T}_k} = \Gamma\left(X^{\mathcal{T}_k}, \mathcal{B}^{\mathcal{T}_k}\right), \quad k = 1, 2, \ldots, 2N. \tag{3}$$

Subsequently, max-pooling $\mathcal{M}(.)$ is employed along the $2N$-aligned feature maps. The concise BEV feature $\mathcal{F}_a^*$ is aggregated as follows:

$$\mathcal{F}_a^* = \mathcal{M}\left(\mathcal{F}_a^{\mathcal{T}_1}, \mathcal{F}_a^{\mathcal{T}_2}, \ldots, \mathcal{F}_a^{\mathcal{T}_{2N}}\right). \tag{4}$$

After the RT BEV pooling process, we employed the RPN to efficiently produce a proposal list $\mathcal{B}^*$ from the lightweight BEV feature $\mathcal{F}_a^*$. These proposals were leveraged to determine the object features for the subsequent stage.

Next, we introduced the RT voxel-pooling method, drawing inspiration from [15], to aggregate the rotational-transformation features. Because the RT backbone produces diverse rotational-transformation features, whereas proposal $\mathcal{B}^*$ is formulated in the original coordinate system T^1, our model requires the proposal to be aligned before pooling.

In our case, it was impossible to directly use a conventional RoI pooling process [8], [10], [12] to extract features from our backbone. Therefore, from the proposal layer $\mathcal{B}^*$ corresponding to the original coordinate, we apply multigrid pooling with size $(6 \times 6 \times 6)$ to obtain a set of RoI grid points. Subsequently, these multiple RoI grid points are transformed into the coordinate systems of each channel within $\{A^{\mathcal{T}_k}\}_{k=1}^{2N}$ by employing transformation actions $\{\mathcal{T}^2, \ldots, \mathcal{T}^{2N}\}$.

For a proposal within $\mathcal{B}^*$, we initially create $2N$ sets of instance-level grid points $\{\mathbf{X}^{\mathcal{T}_k}\}_{k=1}^{2N}$, with each set represented as $\mathbf{X}^{\mathcal{T}_k} = \{X_m^{\mathcal{T}_k}\}_{m=1}^{M}$, where M indicates the number of grid points contained in each set. These grid points were generated based on the rotational-transformation actions $\{\mathcal{T}_k\}_{k=1}^{2N}$.

We then apply voxel-set abstraction $\vartheta(\cdot)$ [8] for voxel neighbor aggregation

$$\mathbf{F}_{\mathbf{P}}^{\mathcal{T}_k} = \vartheta\left(\mathbf{X}^{\mathcal{T}_k}, \mathcal{V}^{\mathcal{T}_k}\right), \quad k = 1, 2, \ldots, 2N. \tag{5}$$

This process yields multiple pooled instance-level features $\{\mathbf{F}_{\mathbf{P}}^{\mathcal{T}_k}\}_{k=1}^{2N} \subset \mathbb{R}^{1\times C}$, where C denotes the number of grid-wise feature channels.

3) Region-Proposal Network: We employed a similar approach [7], [8] for the RPN design to generate high-quality 3-D proposals. The proposed bounding boxes effectively offer information regarding the positions and orientations of the objects in the subsequent stage. First, the RPN condenses the 3-D feature volume by stacking it along the z-axis and subsequently applies a series of 2-D convolutions to the BEV feature maps. The proposals contain anchor classification and regression of the object size, location, and orientation angles with respect to the ground-truth bounding boxes.

Specifically, we adopted intersection over union (IoU)-based matching to assign ground-truth bounding boxes to anchors, following [8]. The anchor configuration is defined as $(l \times w \times h)$, where l, w, and h are the length, width, and height of the bounding boxes, respectively. We used the common setting, which is $(3.9 \times 1.6 \times 1.56)$, $(0.8 \times 0.6 \times 1.73)$, and $(1.76 \times 0.6 \times 1.73)$ for car, pedestrian, and cyclist objects, respectively.

The RPN loss function is formulated as follows:

$$\mathcal{L}_{\text{RPN}} = \frac{1}{N_p}\left[\sum_i \mathcal{L}_{\text{score}}(\alpha_i, \hat{\alpha}_i) + \gamma(\text{IoU}_i > u)\sum_i \mathcal{L}_{\text{reg}}(\delta_i, \hat{\delta}_i)\right] \tag{6}$$

where $\mathcal{L}_{\text{score}}$, $\mathcal{L}_{\text{reg}}$, α_i, $\hat{\alpha}_i$, δ_i, and $\hat{\delta}_i$ denote the smooth $L1$ loss, binary cross-entropy loss, score prediction, score target, residual prediction, and residual target, respectively. Note that the regression loss is computed only for object proposals with $(\text{IoU}_i > u)$.

C. Stage 2: 3-D Object-Shape Enhancement

To enhance the 3-D detector's performance, we introduced a deep-learning network called a VPST. VPST is transformer-based and can reconstruct a complete shape from partial observations, effectively addressing issues related to occlusion. Our autoregressive approach aims to learn the distribution $p(\mathbf{X})$ from the partial observation of 3-D shapes $\mathbf{X}$ to infer the complete shape $\mathbf{Y}$, as illustrated in Fig. 2.

First, from the object features of the previous stage, we split the partial object shape into patches using a patch-wise encoder [34]. This enables the independent encoding of the local context with partial observations. Our goal is to downscale the high-dimensional persistent 3-D form into a discrete latent space to train an efficient autoregressive model. Therefore, we convert the patches of the object shape into a volumetric grid space G with a resolution R using a sequence grid function. Such a grid space and dictionary allow for the efficient modeling of global dependencies by transformers that allow forms to be represented as compact sequences of entry indices, describing the local shapes inside all nonempty grid elements.

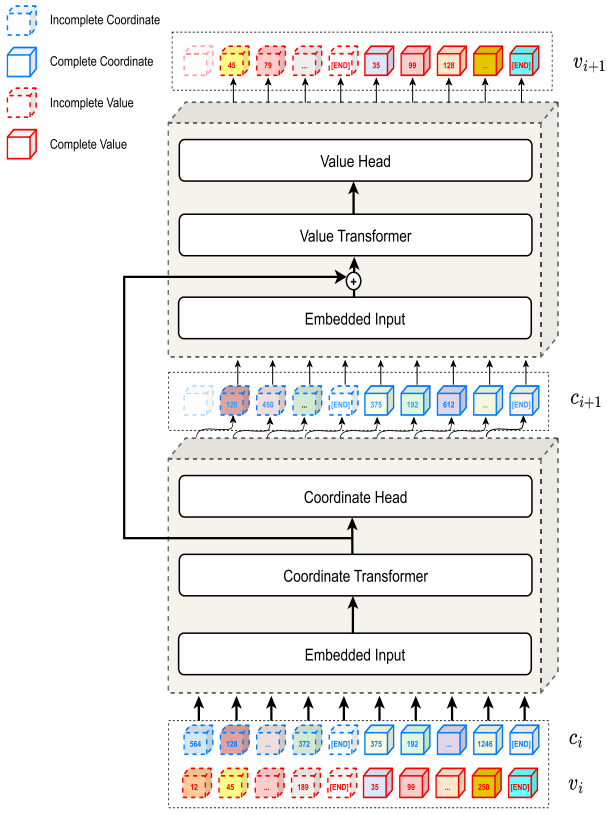


Fig. 3. Architecture of the SCT includes two transformer blocks: the coordinate transformer and value transformer. The discrete tuple pair contains a partial sequence (represented by dashed boxes) and the complete sequence (represented by solid boxes). Both sequences include an appended end token. These sequences are concatenated, and their locations (c_i, enclosed in a blue border) and values (v_i, enclosed in a red border) are fed to a coordinate transformer to predict the subsequent location (c_{i+1}). The value transformer utilizes both c_{i+1} and the previous transformer's output features to predict the next value (v_{i+1}).

We used $R = 64$ resolution for the first grid space $(64 \times 64 \times 64)$, and these cube features were then downsampled to a lower-dimensional discrete space $(32 \times 32 \times 32)$ using a downsampler. Subsequently, the nonempty features were reshaped into a sequence of length K in row-major order. These features are sparse; thus, we use a flattened index $\{c_i\}_{i=0}^{K-1}$ to capture their coordinates.

We reduced the bit size of the feature sequence $\{\mathbf{z}_i\}_{i=0}^{K-1}$ through vector quantization followed by mapping onto the nearest element in the dictionary D of V embeddings $\{\mathbf{e}_j\}_{j=0}^{V}$. The index values of these entries were calculated as follows:

$$v_i = \text{argmin}_j||\mathbf{z}_i - \mathbf{e}_j||_2, \quad j \in [0, V). \tag{7}$$

Subsequently, a discrete tuple pair of a compact sequence for the object shape $\mathcal{S} = \{(c_i, v_i)\}_{i=0}^{K-1}$ was produced. To predict the distribution of the next element conditioned on previous elements using the autoregressive model, we designed a shape-completion transformer (SCT) module, as shown in Fig. 3. Specifically, we stacked two transformer blocks to predict p_{c_i} and p_{v_i}, following [33]. The distribution of each

entry in the tuple is calculated as follows:

$$p(\mathcal{S}_{\mathbf{C}}|\mathcal{S}_{\mathbf{P};\phi}) = \prod_{i=0}^{K-1} p_{c_i} \times p_{v_i}$$
$$p_{c_i} = p(c_i|\mathbf{c}_{<i}, \mathbf{v}_{<i}, \mathcal{S}_{\mathbf{P}}; \phi)$$
$$p_{v_i} = p(v_i|\mathbf{c}_{\leq i}, \mathbf{v}_{<i}, \mathcal{S}_{\mathbf{P}}; \phi) \quad (8)$$

where $\mathcal{S}_{\mathbf{C}}$, $\mathcal{S}_{\mathbf{P}}$, ϕ, p_{c_i}, and p_{v_i} are the complete sequence, partial sequence, model parameters, distribution of coordinates, and index value, respectively. In our case, the index value p_{v_i} depends on the current coordinate c_i. To facilitate the learning strategy, an extra end token was attached to both sequences. The loss function of the SCT was calculated as follows:

$$\mathcal{L}_{\text{SCT}} = -\log p(\mathcal{S}_{\mathbf{C}}|\mathcal{S}_{\mathbf{P}}; \phi). \quad (9)$$

After obtaining $\mathcal{S}_{\mathcal{C}}$, an upsampler and decoder were used. Initially, the quantized sparse sequence is mapped onto a 3-D feature grid. The decoder uses this feature to infer a large map. To help the model better grasp the global knowledge of the object shapes from partial observations, our decoder is composed of multiple ResNet blocks, following [33]. This allows the model to comprehend the distributions that could exist both within and without the object shape.

D. Stage 3: Attention Fusion and Refinement Network

To aggregate all the features from the different domains, we employed a collection of different layers for proposal refinement. The features of the different phases were regularly concatenated in a simple operation. Nonetheless, determining the relationship of features among multiple stages is challenging because concatenating them without additional processing can lead to interference.

Motivated by recent attention methods [39], [43], we present an attention-based mechanism to facilitate the combination of proposal features from various stages. Given a region proposal $\mathcal{B}^*$, we apply the RT voxel-pooling module to extract the instance-level features of partial shape $\mathbf{F}_{\mathbf{P}}^{\mathcal{T}_k} = \{F_m^{\mathcal{T}_k}\}_{m=1}^{M} \subset \mathbb{R}^{1\times C}$, as mentioned in Section III-B. Simultaneously, we acquire a feature representing the complete shape, which is denoted as $\mathbf{F}_{\mathbf{C}}^{\mathcal{T}_k}$.

As depicted in Fig. 4, we first concatenate the instance-level features of partial and complete shapes corresponding to rotational ordering $\mathbf{F}^{\mathcal{T}_k} = [\mathbf{F}_{\mathbf{P}}^{\mathcal{T}_k}, \mathbf{F}_{\mathbf{C}}^{\mathcal{T}_k}]$. This combined feature is then input into the AFM, which helps determine the significance of both the pooled and shape-information features. For each rotational-transformation feature (blue, pink, or orange), the AFM employs channel-wise and point-wise max-pooling layers, allowing features to be extracted from various dimensions to capture different perspectives. These features are subsequently processed using two fully connected layers followed by the rectified linear unit (ReLU) activation function. Following this, matrix multiplication is used to consolidate the channel-wise and point-wise attention maps. The attention values are normalized using a sigmoid function. Finally, an attention map is employed to enhance the concatenated features through element-wise multiplication. These combined features $\hat{\mathbf{F}}^{\mathcal{T}_k}$ serve as inputs to the attention-refinement module (ARM), which generates the final precise result.

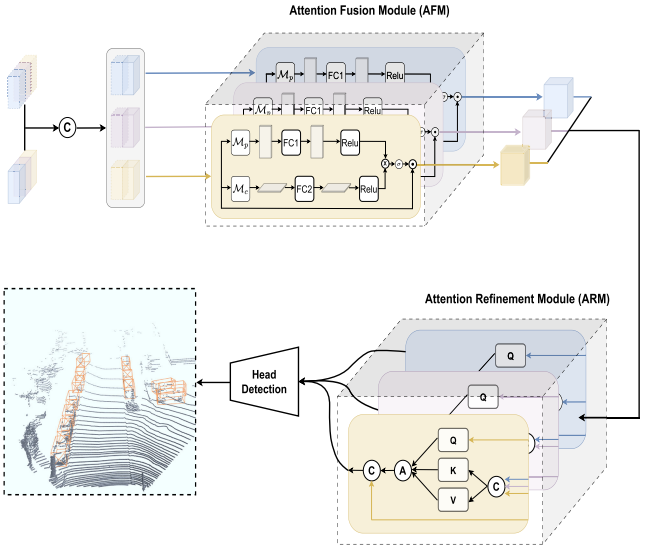


Fig. 4. Illustration of the AFR network. The AFM aggregates all features from the preceding stage, which are then refined by the ARM to produce the final precise predictions. The **C**, **A**, $\times$, σ, and $\cdot$ denote the concatenation operation, attention operation, matrix multiplication, sigmoid function, and element-wise multiplication, respectively.

In the ARM, we gather the encoded features from all previous layers and the current layer as $\hat{\mathbf{F}}^j = [\hat{\mathbf{F}}^{\mathcal{T}_0}, \hat{\mathbf{F}}^{\mathcal{T}_1}, \ldots, \hat{\mathbf{F}}^{\mathcal{T}_j}]$, j $\in$ (0, k). Subsequently, we project these features linearly to obtain $\mathbf{Q}^j = \hat{F}^j\mathbf{W}_q^j$, $\mathbf{K}^j = \hat{F}^j\mathbf{W}_k^j$, and $\mathbf{V}^j = \hat{F}^j\mathbf{W}_v^j$, which represent the query, key, and value embeddings, respectively. To effectively capture the relationship between the various rotational layers, we employed multihead attention. The attention value of each attention head i is

$$\hat{\mathbf{F}}_i^j = \text{softmax}\left(\frac{\mathbf{Q}_{\text{i}}^{\text{j}}\left(\mathbf{K}_{\text{i}}^{\text{j}}\right)^{\text{T}}}{\sqrt{C'}}\right)\mathbf{V}_{\text{i}}^{\text{j}} \quad (10)$$

where C' is the channel size in the multihead attention. Subsequently, the box-voting method [39] was adopted to directly standardize the confidence prediction and fuse the bounding boxes

$$C = \frac{1}{N_r}\sum_j C^j \quad (11)$$

$$B = \frac{1}{\sum_j C^j}\sum_j C^j \cdot B^j. \quad (12)$$

Here, C and B represent the merged confidence prediction and bounding boxes, respectively. Following the box-voting process, we obtain a collection of refined, high-quality boxes. To eliminate redundancies, we conduct nonmaximum suppression (NMS) on the voted results to generate the final detection outputs. This voting mechanism enables the integration of diverse predictions from different refiners, leading to more accurate and reliable final predictions.

E. Overall Loss Function

The proposed TSSTDet is trained in an end-to-end manner. Our overall loss function includes L_{RPN} of the RPN, L_{SCT} of

the SCT, and L_{AFR} of the AFR network. L_{AFR} is the sum of multiple refinement losses in multiple layers, as mentioned in Section III-B3.

Within each refinement-attention layer, we incorporate the box regression loss $\mathcal{L}_{\text{reg}}$ and score loss $\mathcal{L}_{\text{score}}$ following [1] and [8]. The loss function for the AFR module is formulated as follows:

$$\mathcal{L}_{\text{AFR}} = \frac{1}{N_b}\left[\sum_i \sum_j \mathcal{L}_{\text{score}}\left(\alpha_i^j, \hat{\alpha}_i^j\right) + \gamma\left(\text{IoU}_i^j > u_i\right)\sum_i \sum_j \mathcal{L}_{\text{reg}}\left(\delta_i^j, \hat{\delta}_i^j\right)\right] \quad (13)$$

where $\mathcal{L}_{\text{score}}$, $\mathcal{L}_{\text{reg}}$, α_i^j, $\hat{\alpha}_i^j$, δ_i^j, and $\hat{\delta}_i^j$ denote the smooth $L1$ loss, binary cross-entropy loss, score prediction, score target, residual prediction, and residual target for the ith proposal at the jth refinement layer, respectively. Note that the regression loss is computed only for object proposals with $(\text{IoU}_i > u)$.

Overall, the total loss $\mathcal{L}_{\text{total}}$ is calculated as follows:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{RPN}} + \mu\mathcal{L}_{\text{SCT}} + \mathcal{L}_{\text{AFR}} \quad (14)$$

where μ is a hyperparameter. We set $\mu = 0.9$ for the best performance.

IV. Experiments

A. Datasets and Metrics

1) KITTI Dataset: The KITTI dataset [61] is a widely used dataset that consists of 7481 LiDAR training frames and 7518 LiDAR testing frames. For our experiment, we split the training data into two sets, one with 3712 frames for training and another with 3769 frames for validation, following a state-of-the-art method [6], [8]. The primary evaluation metric was the 3-D average precision (AP) calculated under 40 different recall thresholds ($R40$) at the easy, moderate, and hard levels. In addition, we report the results on the validation set using a 3-D AP metric under 11 recall thresholds for the moderate car class, following the conventions set by previous studies [2], [8]. The specified IoU thresholds for this metric are 0.7 for cars, 0.5 for pedestrians, and 0.5 for cyclists.

2) Waymo Open Dataset: The Waymo open dataset (WOD) [62] includes 798, 202, and 150 sequences for training, validation, and testing, respectively. Our model's performance was assessed using the following official metrics: 3-D mean AP (mAP) and 3-D mAP weighted by heading (mAPH) for all categories. The LEVEL 1 mAP computes the mAP for classes with more than five points, whereas the LEVEL 2 mAP computes the mAP for classes with at least one point. It is worth noting that the primary ranking metric for the Waymo 3-D detection challenge is mAPH ($L2$), with IoU thresholds of 0.7, 0.5, and 0.5 for vehicles, pedestrians, and cyclists, respectively.

B. Setup Details

For the KITTI dataset, our detection range, number of proposals, and NMS threshold were consistent with those used in the baseline detectors, Voxel-RCNN [7], and TED [6]. The voxel size configuration was (0.05, 0.05, and 0.1 m), and we set the range of the point cloud to be [0, 70.4], $[-40, 40]$, and $[-3, 1]$ m for the X-, Y-, and Z-axes, respectively. We configured the number of rotations as $N = 3$, and the number of multi-head attentions as $j = 3$.

Our TSSTDet model was trained for 40 epochs using the ADAM optimizer. The learning rate, weight decay, and momentum were 0.01, 0.01, and 0.9, respectively. In detail, we configure the momentum schedule to decrease from 0.95 to 0.85 and decay the learning rate from epoch 35 to epoch 40 during training. The learning rate decay, percentage of total epochs for rising learning rate, and the gradient clipping threshold are 0.1, 0.4, and 10, respectively. We utilized an NMS threshold of 0.8 to produce 160 RoI proposals with an equal ratio of positive and negative samples. For the testing phase, after proposal refinement, we adjusted the threshold to 0.1 to eliminate redundant boxes. With regard to data augmentation, our approach performed excellently, even without incorporating rotation and reflection data-augmentation techniques. Similarly, for the Waymo dataset [62], we adopted the same configuration as that of PV-RCNN [8]. Throughout the experiments, TSSTDet was trained on two NVIDIA GeForce RTX 3090 GPUs with a batch size of 4.

V. Results

This section presents a comparative analysis of our proposed model against state-of-the-art methods, utilizing the KITTI and WOD datasets. For the KITTI offline evaluation, our model was exclusively trained on the 3712-sample training set, and the results were subsequently reported on the validation set. Our model outperformed all the other models on the KITTI validation set across all classes. To assess our model's performance on the KITTI test set, we submitted the results to the KITTI online benchmark server. Our TSSTDet model holds the top rank on the KITTI online benchmark, particularly for the car category at https://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=3d. For the WOD dataset, we evaluated the TSSTDet using a validation set to demonstrate our results.

A. Evaluation

1) KITTI Test Set: We uploaded our test set results to the KITTI online server in compliance with KITTI regulations (only the best results were submitted to the online test server). The comparison between our proposed framework and state-of-the-art models on the KITTI test set is shown in Table I. We reported the evaluation on the most important car category. Note that "Mod." indicates the moderate difficulty level. On the KITTI leaderboard, a moderate AP of under 40 recalls is the official ranking metric.

We surpassed all state-of-the-art methods that used only LiDAR point clouds on both single-stage and multistage. Our results outperformed other LiDAR-only methods on 3-D detection, with scores of 91.84%, 85.47%, and 80.65% for the easy, moderate, and hard difficulty levels, respectively.

TABLE I
COMPARISON OF ALL 3-D DETECTOR RESULTS ON THE KITTI TEST SET. WE EVALUATE OUR MODEL FOR THE CAR CLASS USING THE 3-D AP UNDER 40 RECALL THRESHOLDS. OUR TSSTDET SURPASSED ALL OF THE STATE-OF-THE-ART METHODS ON THE 3-D OBJECT DETECTION BENCHMARK. THE BEST PERFORMANCES ARE HIGHLIGHTED IN BOLD

Model	Reference	Stage	Car 3-D$_{R40}$			Car BEV$_{R40}$		
			Easy	Mod.	Hard	Easy	Mod.	Hard
LiDAR + RGB								
MV3D [22]	CVPR 2017	Multiple	74.97	63.63	54.00	86.62	78.93	69.80
ContFuse [44]	ECCV 2018	Single	83.68	68.78	61.67	94.07	85.35	75.88
F-PointNet [23]	CVPR 2018	Multiple	82.19	69.79	60.59	91.17	84.67	74.77
AVOD [26]	IROS 2018	Multiple	83.07	71.76	65.73	90.99	84.82	79.62
F-ConvNet [45]	IROS 2019	Multiple	87.36	76.39	66.69	91.51	85.84	76.11
3D-CVF [46]	ECCV 2020	Multiple	89.20	80.05	73.11	93.52	89.56	82.45
CLOC [47]	IROS 2020	Multiple	88.94	80.67	77.15	93.05	89.80	86.57
SFD [27]	CVPR 2022	Multiple	91.73	84.76	77.92	95.64	91.85	86.83
PA3DNet [48]	TII 2023	Multiple	90.49	82.57	77.88	93.11	89.46	84.60
LoGoNet [49]	CVPR 2023	Multiple	91.80	85.06	**80.74**	95.48	91.52	87.09
TED [6]	AAAI 2023	Multiple	91.61	85.28	80.68	95.44	92.05	87.30
LiDAR Only								
VoxelNet [2]	CVPR 2018	Single	77.82	64.17	57.51	87.95	78.39	71.29
SECOND [1]	Sensors 2018	Single	83.13	73.66	66.20	88.07	79.37	77.95
PointRCNN [10]	CVPR 2019	Multiple	86.96	75.64	70.70	92.13	87.39	82.72
TANet [50]	AAAI 2020	Multiple	84.39	75.94	68.82	91.58	86.54	81.19
Part-A^2 [51]	TPMAI 2020	Multiple	87.81	78.49	73.51	91.70	87.79	84.61
3DSSD [21]	CVPR 2020	Single	88.36	79.57	74.55	92.66	89.02	85.86
SA-SSD [3]	CVPR 2020	Single	88.75	79.79	74.16	95.03	91.03	85.96
CIA-SSD [52]	CVPR 2020	Single	89.59	80.28	72.87	93.74	89.84	82.39
PV-RCNN [8]	CVPR 2020	Multiple	90.25	81.43	76.82	94.98	90.65	86.14
CT3D [41]	ICCV 2021	Multiple	87.83	81.77	77.16	92.36	88.83	84.07
Voxel-RCNN [7]	AAAI 2021	Multiple	90.90	81.62	77.06	94.85	88.83	86.13
SPG [53]	ICCV 2021	Multiple	90.64	82.66	77.91	92.80	89.12	86.27
SE-SSD [4]	CVPR 2021	Single	91.49	82.54	77.15	95.68	91.84	86.72
VoxSeT [54]	CVPR 2022	Single	88.53	82.06	77.46	92.70	89.07	86.29
BADet [55]	PR 2022	Multiple	89.28	81.61	76.58	95.23	91.32	86.48
RDIoU [56]	ECCV 2022	Single	90.65	82.30	77.26	94.90	89.75	84.67
BtcDet [12]	AAAI 2022	Multiple	90.64	82.86	78.09	92.81	89.34	84.55
CASA [39]	TGRS 2022	Multiple	91.58	83.06	80.08	94.57	91.22	88.43
GLENet-VR [57]	IJCV 2023	Multiple	91.67	83.23	78.43	93.48	89.76	84.89
PVT-SSD [58]	CVPR 2023	Single	90.65	82.29	76.85	95.23	91.63	86.43
3D HANet [59]	TGRS 2023	Multiple	90.79	84.18	77.57	94.33	91.13	86.33
OcTr [60]	CVPR 2023	Multiple	90.88	82.64	77.77	93.08	89.56	86.74
TSSTDet (Ours)	-	Multiple	**91.84**	**85.47**	80.65	**95.80**	**92.11**	**89.23**

Specifically, for the most important evaluation metric of car moderate, we achieved the top position in the rankings for both LiDAR + RGB and LiDAR-only methods. Furthermore, we achieved the top rank for car BEV$_{R40}$ levels, with scores of 95.80%, 92.11%, and 89.23% for the easy, moderate, and hard difficulty levels, respectively.

2) KITTI Validation Set: We also reported an evaluation using the KITTI validation dataset, as listed in Table II. In the table, "Mod." represents the moderate level, while "$*$" indicates that the results are reproduced from the open-source code [65]. Our model consistently outperformed all existing LiDAR-based methods across all classes, including cars, pedestrians, and cyclists.

Notably, in the crucial category of Car 3-D AP$_{R40}$, TSSTDet demonstrated substantial improvements over the state-of-the-art TED-S method [6], with margins of 2.24%, 1.09%, and 1.11% for the easy, moderate, and hard difficulty levels, respectively. We also excelled in the moderate Car 3-D AP$_{R11}$ category, confirming our strong performance on the leaderboard.

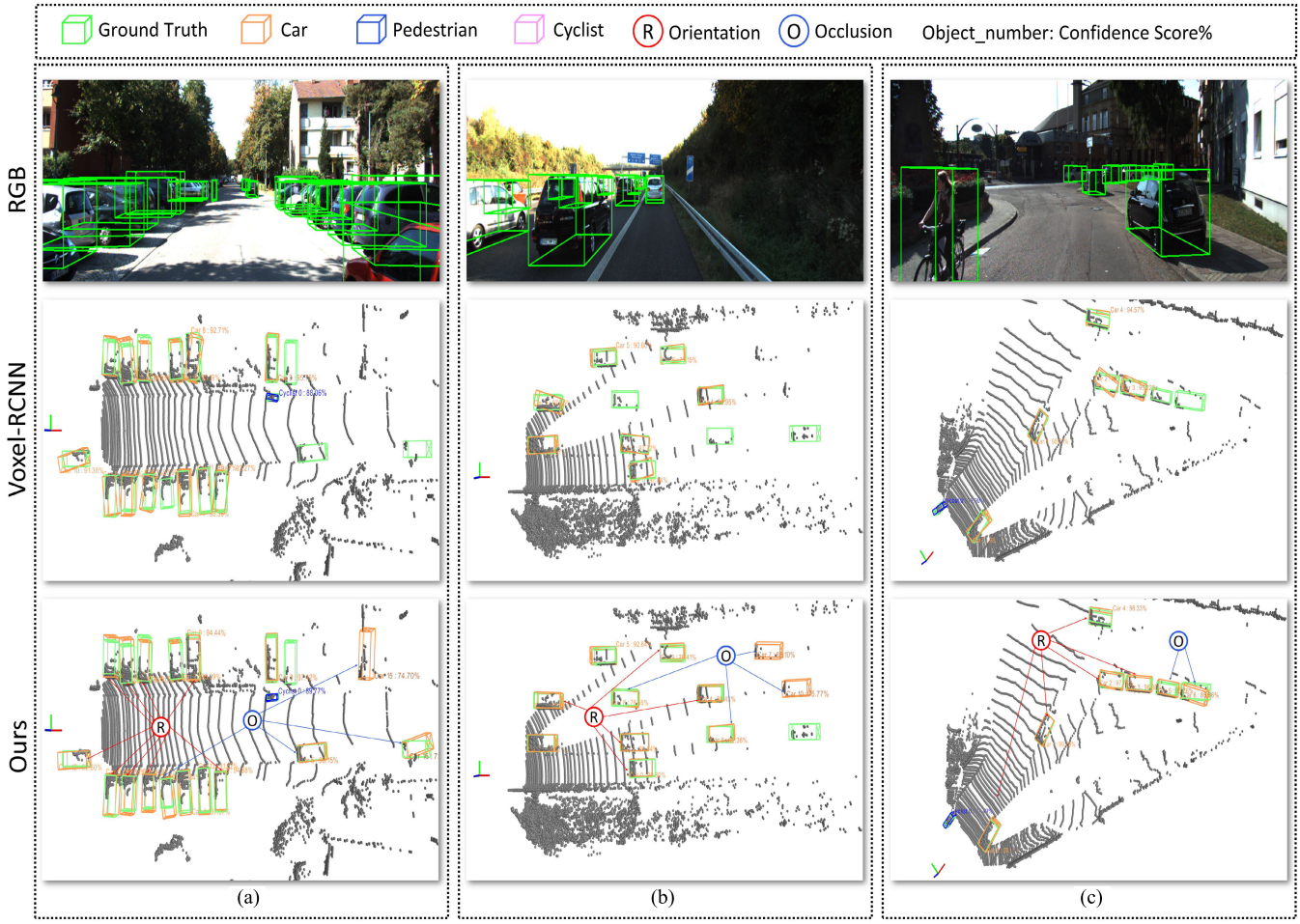


Fig. 5. Visual assessment of TSSTDet using the KITTI validation dataset. Columns (a)–(c) represent the various scenarios. In each scenario, the initial row exhibits the RGB image along with green ground-truth annotations for all classes. The second and third rows depict the LiDAR frame with predicted results of the Voxel-RCNN baseline and our model, respectively. Predicted 3-D bounding boxes in LiDAR frames of cars, pedestrians, and cyclists are represented in orange, pink, and blue, respectively. ***R*** and ***O*** denote cases where our model surpasses the baseline in terms of orientation and occlusion accuracy, respectively. Best viewing experience through color and zoom functionality.

TABLE II

COMPARISON OF ALL 3-D DETECTOR RESULTS ON THE KITTI VALIDATION SET. WE EVALUATED OUR MODEL FOR ALL THREE CLASSES, USING THE 3-D AP UNDER 40 RECALL THRESHOLDS. WE ALSO REPORT THE AP UNDER 11 RECALL THRESHOLDS FOR THE MODERATE CAR CLASS. OUR TSSTDET SURPASSED ALL STATE-OF-THE-ART METHODS ON THE 3-D OBJECT DETECTION BENCHMARK. THE BEST PERFORMANCES ARE HIGHLIGHTED IN BOLD, AND ∗ INDICATES RESULTS REPRODUCED FROM OPEN-SOURCE CODE

Model	Reference	Modality	Car 3D$_{R40}$			Pedestrian 3D$_{R40}$			Cyclist 3D$_{R40}$			Car 3D AP$_{R11}$
			Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Mod.
SA-SSD* [3]	CVPR 2020	LiDAR	92.49	84.53	81.71	61.72	55.01	49.94	87.82	71.25	65.88	79.91
PV-RCNN [8]	CVPR 2020	LiDAR	92.57	84.83	82.69	64.26	56.67	51.91	88.88	71.95	66.78	83.90
Voxel R-CNN* [7]	AAAI 2021	LiDAR	92.15	85.19	82.78	64.88	57.32	52.11	89.25	72.52	67.03	84.52
CT3D [41]	ICCV 2021	LiDAR	92.85	85.82	83.46	65.73	58.56	53.04	91.99	71.60	67.34	86.06
SE-SSD* [4]	CVPR 2021	LiDAR	93.19	86.12	83.31	67.98	59.72	54.83	91.77	72.54	68.78	85.71
BtcDet [12]	AAAI 2022	LiDAR	93.15	86.28	83.86	69.39	61.19	55.86	91.45	74.70	70.08	86.57
CASA [39]	TGRS 2022	LiDAR	93.38	86.42	84.04	68.81	62.59	57.47	92.81	72.63	68.32	86.63
GLENet [57]	IJCV 2023	LiDAR	93.51	86.10	83.60	69.55	64.12	59.23	92.77	72.44	68.11	86.46
TED-S [6]	AAAI 2023	LiDAR	93.05	87.91	85.81	72.38	67.81	63.54	93.09	75.77	71.20	87.54
TSSTDet (Ours)	-	LiDAR	**95.29**	**89.06**	**86.92**	**75.13**	**69.38**	**64.31**	**95.16**	**76.24**	**71.62**	**88.15**

It is worth mentioning that our model exhibited excellent results for the pedestrian and cyclist classes. Specifically, TSSTDet outperformed the most recent TED model [6] on the moderate level by 1.57% and 0.47% for pedestrians and

TABLE III
COMPARISON OF THE BEV OBJECT DETECTION ON THE VALIDATION SET FOR THE CAR CLASS. THE BEST PERFORMANCES ARE HIGHLIGHTED IN BOLD. OUR TSSTDET SURPASSED ALL EXISTING STATE-OF-THE-ART METHODS

Model	Modality	$AP_{R40}(IoU = 0.7)$		
		Easy	Mod.	Hard
SIENet [11]	LiDAR	90.29	88.41	87.77
PV-RCNN [8]	LiDAR	95.76	91.11	88.93
Voxel-RCNN [7]	LiDAR	95.52	91.25	88.99
CT3D [41]	LiDAR	96.14	91.88	89.63
SE-SSD [4]	LiDAR	96.59	92.28	89.72
VPFNet [63]	LiDAR + RGB	94.11	92.44	89.88
Graph R-CNN [64]	LiDAR + RGB	96.28	92.68	92.11
TSSTDet (Ours)	LiDAR	**97.02**	**94.36**	**92.69**

cyclists, respectively. We also conducted experiments on BEV object detection for the car class. The results are shown in Table III, with "Mod." indicating a moderate level. Our TSSTDet surpassed all previous methods with different modalities and obtained the highest scores for the car class of 97.02%, 94.36%, and 92.69% for the easy, moderate, and hard difficulty levels, respectively.

Additionally, Fig. 5 provides the visual assessment of our model on the KITTI validation set. The substantial performance improvements are primarily attributed to the rotational-transformation and shape-completion design, which enhance the model's ability to capture object-shape features, resulting in superior detection performance.

3) Waymo Validation Set: We conducted a comparative analysis of TSSTDet and other models using the WOD. Our approach is consistent with the usage of a single frame, as observed in Voxel-RCNN [7] and PV-RCNN [8]. The results are summarized in Table IV, where "N/A" indicates that certain information was not available.

To evaluate vehicle detection, we used both the 3-D mAP and 3-D mAP weighted by heading (mAPH). In all categories, our model consistently outperformed all state-of-the-art detectors. Specifically, we outperformed all state-of-the-art methods in the most important vehicle (*L2*) categories, achieving scores of 71.75% and 70.12% for mAP and mAPH, respectively.

Overall, TSSTDet showed substantial improvements for Level 2 objects across all categories. This can be attributed to the fact that objects with fewer data points are more likely to lack shape and object-orientation information. These results for the WOD, which is one of the largest publicly available LiDAR datasets, underscore the effectiveness of our model.

B. Ablation Study

We conducted a series of experiments using various configurations to analyze the effectiveness of each TSSTDet module. The number of hyperparameters in the model was carefully examined using various metrics. We used the KITTI dataset and open-source code [65] to reproduce the baseline results.

1) Effectiveness of RT Backbone and Pooling: The number of rotational transformations N of TSSTDet is a hyperparameter. It affects the performance of the backbone and is also related to the number of attention layers in the AFR. To evaluate the impact of parameter N and determine the optimal hyperparameter, we conducted a set of ablation experiments on the car, pedestrian, and cyclist classes using the KITTI validation dataset, as shown in Table V.

When $N = 3$, the performance showed a substantial enhancement in comparison to using a single rotation number, with notable improvements of 2.72%, 2.82%, and 1.61% for the car, pedestrian, and cyclist classes, respectively. When comparing $N = 4$ with $N = 3$, it is evident that employing $N = 3$ maintains an excellent performance while maintaining an optimal processing speed (running on a single RTX 3090 GPU). Therefore, we chose $N = 3$ as the preferred setting for our primary model to achieve efficient performance.

The effectiveness of the RT backbone and pooling is reported in Table VI. We used Voxel-RCNN [7] as the baseline. As depicted in the third row, there is a notable 0.66% enhancement in the performance of the car class at a moderate level, compared to the baseline.

2) Effectiveness of VPST: Next, we comprehensively verified the contribution of the VPST module to the model performance under AP_{R40} of the car category. We utilized open-source code [65] to examine the results of various configurations. The VPST module aims to reconstruct the complete shape of an object, which contributes to the performance enhancement.

As illustrated in the second and fourth rows of Table VI, our VPST boosts the AP 3-D detection by 0.58% and 0.57% compared to the baseline, respectively. The metric used is the moderate level of the KITTI car category in 3-D detection. When VPST was combined with all the other modules, the best performance was achieved at 89.06%.

Additionally, we explored the effectiveness of VPST with respect to the orientation aspect, as shown in Table VII. In Table VII, we present a different setting by extending our module to a TED-S-based detector [6]. The second row demonstrates that our VPST leads to a better performance at all three levels of average orientation similarity (AOS), compared to the baseline.

3) Effectiveness of AFR: To assess the impact of AFR, we first built a baseline Voxel-RCNN [7]. The comparative results are presented in Table VI. Employing a simple head detector did not improve the performance. However, upon incorporating our AFR, the performance was further enhanced to 89.06%, as shown in the last row. Our AFR effectively aggregated object-shape features from various angles, resulting in more robust detections from sparse points.

To further validate the effectiveness of our AFR, we conducted additional experiments on the KITTI dataset using the orientation metric. As indicated in the third row of Table VII, our AFR delivers impressive performance improvements in terms of easy (0.33%), moderate (0.67%), and hard (0.4%)

TABLE IV
COMPARISON OF ALL STATE-OF-THE-ART 3-D DETECTION PERFORMANCES ON THE WAYMO VALIDATION SET. *L1* AND *L2* REFER TO LEVELS 1 AND 2, RESPECTIVELY. THE BEST PERFORMANCES ARE HIGHLIGHTED IN BOLD

Model	Vehicle (L1)		Vehicle (L2)		Pedestrian (L1)		Pedestrian (L2)		Cyclist (L1)		Cyclist (L2)	
	mAP	mAPH	mAP	mAPH	mAP	mAPH	mAP	mAPH	mAP	mAPH	mAP	mAPH
BtcDet [12]	78.58	78.06	70.10	69.61	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
PV-RCNN [8]	78.79	78.21	70.26	69.71	76.67	67.15	68.51	59.72	68.98	67.63	66.48	65.17
CASA [39]	78.54	78.00	69.91	69.42	80.88	73.10	71.87	64.78	69.66	68.38	67.07	66.83
TED [6]	79.26	78.73	70.50	70.07	82.62	76.66	73.50	68.03	74.11	72.94	71.46	70.32
TSSTDet (Ours)	**80.37**	**79.98**	**71.75**	**70.12**	**83.07**	**77.34**	**73.55**	**69.15**	**74.25**	**73.13**	**72.44**	**71.16**

TABLE V
EFFECT OF RTCONV USING DIFFERENT ROTATION NUMBERS

Rotation number (N)	3-D Detection (Mod.)			FPS
	Car	Pedestrian	Cyclist	
1	86.34	66.56	74.63	18.2
2	88.54	67.47	75.01	13.9
3	89.06	**69.38**	**76.24**	10.5
4	**89.09**	68.72	75.11	7.7

TABLE VI
PERFORMANCE ANALYSIS ON THE KITTI VALIDATION SET USING DIFFERENT DESIGNED COMPONENTS. BASELINE, RTCONV, RT POOLING, VPST, AND AFR DENOTE THE VOXEL-RCNN BASELINE, ROTATIONAL-TRANSFORMATION CONV, ROTATIONAL-TRANSFORMATION POOLING, VPST, AND AFR, RESPECTIVELY

Method	Backbone	RT Pooling	VPST	AFR	Mod.
Baseline	SpConv				87.66
TSSTDet_1	SpConv		✓		88.24
TSSTDet_2	RTConv	✓			88.32
TSSTDet_3	RTConv	✓	✓		88.89
TSSTDet_4	RTConv	✓	✓	✓	**89.06**

TABLE VII
ORIENTATION ACCURACY FOR THE KITTI VALIDATION SET

Method	AOS@Easy	AOS@Moderate	AOS@Hard
Basedline	98.39	96.05	93.57
Basedline+VPST	99.24	97.22	94.93
Basedline+VPST+AFR	**99.57**	**97.89**	**95.33**

AOS, compared with the second row. Note that the baseline in Table VII was based on the TED model.

4) Memory Usage and Runtime Analysis: We also assess the model's performance in terms of memory usage and runtime, as shown in Table VIII. For simplicity, we conducted the experiments on a single RTX 3090 with a batch size of 1.

TABLE VIII
MEMORY USAGE AND RUNTIME ANALYSIS FOR EACH MODULE. S AND GB DENOTE SECOND AND GIGABYTE, RESPECTIVELY

Module	Backbone	RPN	VPST	AFR	Total
Runtime (s)	0.039	0.011	0.032	0.015	0.097
Memory usage (GB)	4.896	0.984	4.395	0.146	10.421

We utilize the default configuration with a rotation number set to 3.

VI. CONCLUSION

Research focused on 3-D object detection in situations involving obstructions is scarce. In this article, we presented a multistage 3-D object detector based on a transformer architecture called TSSTDet. To fully leverage the rotational-transformation features, an RT backbone was devised to extract the general pattern. Furthermore, TSSTDet addressed instances with missing object shapes through an autoregression module, VPST, which helped the model comprehensively identify object shapes, even in occluded scenarios. Finally, an effective aggregation and refinement strategy, AFR, was applied to fine-tune the precise prediction. Our model is not simply a high-performance 3-D object detector relying solely on LiDAR point clouds; it also demonstrates remarkable adaptability in scenarios with occluded objects and diverse rotation situations. The experimental results obtained from both the KITTI and Waymo datasets demonstrated the effectiveness and adaptability of the proposed approach. In our opinion, this approach has great potential for a variety of downstream 3-D applications, such as object tracking and motion planning. Future research will concentrate on developing methods that can more effectively aggregate proposals for small objects, such as cyclists and pedestrians.

REFERENCES

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

[2] 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.

[3] C. He, H. Zeng, J. Huang, X.-S. Hua, and L. Zhang, "Structure aware single-stage 3D object detection from point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11870–11879.

[4] W. Zheng, W. Tang, L. Jiang, and C.-W. Fu, "SE-SSD: Self-ensembling single-stage object detector from point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 14489–14498.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.

[6] H. Wu, C. Wen, W. Li, X. Li, R. Yang, and C. Wang, "Transformation-equivariant 3D object detection for autonomous driving," in *Proc. AAAI Conf. Artif. Intell.*, vol. 37, 2023, pp. 2795–2802.

[7] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, and H. Li, "Voxel R-CNN: Towards high performance voxel-based 3D object detection," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 2, 2021, pp. 1201–1209.

[8] 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.

[9] H. A. Hoang and M. Yoo, "3ONet: 3-D detector for occluded object under obstructed conditions," *IEEE Sensors J.*, vol. 23, no. 16, pp. 18879–18892, Aug. 2023.

[10] 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.

[11] Z. Li, Y. Yao, Z. Quan, W. Yang, and J. Xie, "SIENet: Spatial information enhancement network for 3D object detection from point cloud," 2021, *arXiv:2103.15396*.

[12] Q. Xu, Y. Zhong, and U. Neumann, "Behind the curtain: Learning occluded shapes for 3D object detection," in *Proc. AAAI Conf. Artif. Intell.*, 2022, vol. 36, no. 3, pp. 2893–2901.

[13] A. Dosovitskiy et al., "An image is worth 16×16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.

[14] H.-X. Yu, J. Wu, and L. Yi, "Rotationally equivariant 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 1446–1454.

[15] J. Han, J. Ding, N. Xue, and G.-S. Xia, "ReDet: A rotation-equivariant detector for aerial object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 2785–2794.

[16] J. Mao et al., "Voxel transformer for 3D object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 3144–3153.

[17] 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.

[18] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on X-transformed points," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, S. Bengio and H. Wallach and H. Larochelle and K. Grauman and N. Cesa-Bianchi and R. Garnett, Eds. Curran Associates, 2018. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2018/file/f5f8590cd58a54e94377e6ae2eded4d9-Paper.pdf

[19] 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.

[20] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia, "STD: Sparse-to-dense 3D object detector for point cloud," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1951–1960.

[21] 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.

[22] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3D object detection network for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6526–6534.

[23] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum PointNets for 3D object detection from RGB-D data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 918–927.

[24] X. Chen, T. Zhang, Y. Wang, Y. Wang, and H. Zhao, "FUTR3D: A unified sensor fusion framework for 3D detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2023, pp. 172–181.

[25] Z. Liu et al., "BEVFusion: Multi-task multi-sensor fusion with unified bird's-eye view representation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2023, pp. 2774–2781.

[26] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3D proposal generation and object detection from view aggregation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1–8.

[27] X. Wu et al., "Sparse fuse dense: Towards high quality 3D detection with depth completion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 5408–5417.

[28] T. Yin, X. Zhou, and P. Krähenbühl, "Multimodal virtual point 3D detection," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 16494–16507.

[29] Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: Point cloud auto-encoder via deep grid deformation," 2017, *arXiv:1712.07262*.

[30] X. Wen, T. Li, Z. Han, and Y.-S. Liu, "Point cloud completion by skip-attention network with hierarchical folding," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1936–1945.

[31] L. P. Tchapmi, V. Kosaraju, H. Rezatofighi, I. Reid, and S. Savarese, "TopNet: Structural point cloud decoder," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 383–392.

[32] P. Xiang et al., "SnowflakeNet: Point cloud completion by snowflake point deconvolution with skip-transformer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 5479–5489.

[33] X. Yan, L. Lin, N. J. Mitra, D. Lischinski, D. Cohen-Or, and H. Huang, "ShapeFormer: Transformer-based shape completion via sparse representation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 6229–6239.

[34] P. Mittal, Y.-C. Cheng, M. Singh, and S. Tulsiani, "AutoSDF: Shape priors for 3D completion, reconstruction and generation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 306–315.

[35] M.-T. Duong, S. Lee, and M.-C. Hong, "DMT-Net: Deep multiple networks for low-light image enhancement based on retinex model," *IEEE Access*, vol. 11, pp. 132147–132161, 2023.

[36] M.-T. Duong and M.-C. Hong, "EBSD-Net: Enhancing brightness and suppressing degradation for low-light color image using deep networks," in *Proc. IEEE Int. Conf. Consum. Electron.-Asia (ICCE-Asia)*, Oct. 2022, pp. 1–4.

[37] A. Toppo and M. Kumar, "A review of generative pretraining from pixels," in *Proc. 3rd Int. Conf. Adv. Comput., Commun. Control Netw. (ICAC3N)*, Dec. 2021, pp. 1691–1703.

[38] D. Zhang, H. Zhang, J. Tang, M. Wang, X. Hua, and Q. Sun, "Feature pyramid transformer," in *Proc. 16th Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 323–339.

[39] 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, pp. 1–11, 2022, Art. no. 5704511, doi: 10.1109/TGRS.2022.3203163.

[40] J. Yang et al., "Modeling point clouds with self-attention and Gumbel subset sampling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3318–3327.

[41] H. Shenga et al., "Improving 3D object detection with channel-wise transformer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 2723–2732.

[42] B. Graham, M. Engelcke, and L. van der Maaten, "3D semantic segmentation with submanifold sparse convolutional networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9224–9232.

[43] J. Li et al., "3D IoU-Net: IoU guided 3D object detector for point clouds," 2020, *arXiv:2004.04962*.

[44] M. Liang, B. Yang, S. Wang, and R. Urtasun, "Deep continuous fusion for multi-sensor 3D object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 641–656.

[45] Z. Wang and K. Jia, "Frustum ConvNet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 1742–1749.

[46] J. H. Yoo, Y. Kim, J. S. Kim, and J. W. Choi, "3D-CVF: Generating joint camera and LiDAR features using cross-view spatial feature fusion for 3D object detection," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, Aug. 2020, pp. 720–736.

[47] S. Pang, D. Morris, and H. Radha, "CLOCs: Camera-LiDAR object candidates fusion for 3D object detection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 10386–10393.

[48] M. Wang, L. Zhao, and Y. Yue, "PA3DNet: 3-D vehicle detection with pseudo shape segmentation and adaptive camera-LiDAR fusion," *IEEE Trans. Ind. Informat.*, vol. 19, no. 11, pp. 10693–10703, Nov. 2023, doi: 10.1109/TII.2023.3241585.

[49] X. Li et al., "LoGoNet: Towards accurate 3D object detection with local-to-global cross-modal fusion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 17524–17534.

[50] Z. Liu et al., "TANet: Robust 3D object detection from point clouds with triple attention," in *Proc. AAAI Conf. Artifi. Intel.*, 2020, vol. 34, no. 7, pp. 11677–11684.

[51] 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*.
[52] W. Zheng, W. Tang, S. Chen, L. Jiang, and C.-W. Fu, "CIA-SSD: Confident IoU-aware single-stage object detector from point cloud," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 4, pp. 3555–3562.
[53] Q. Xu, Y. Zhou, W. Wang, C. R. Qi, and D. Anguelov, "SPG: Unsupervised domain adaptation for 3D object detection via semantic point generation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 15426–15436.
[54] C. He, R. Li, S. Li, and L. Zhang, "Voxel set transformer: A set-to-set approach to 3D object detection from point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 8407–8417.
[55] R. Qian, X. Lai, and X. Li, "BADet: Boundary-aware 3D object detection from point clouds," *Pattern Recognit.*, vol. 125, Mar. 2022, Art. no. 108524.
[56] H. Sheng et al., "Rethinking IoU-based optimization for single-stage 3D object detection," in *Computer Vision—ECCV 2022*. Cham, Switzerland: Springer, 2022, pp. 544–561.
[57] Y. Zhang, Q. Zhang, Z. Zhu, J. Hou, and Y. Yuan, "GLENet: Boosting 3D object detectors with generative label uncertainty estimation," *Int. J. Comput. Vis.*, vol. 131, no. 12, pp. 3332–3352, Dec. 2023.
[58] H. Yang et al., "PVT-SSD: Single-stage 3D object detector with point-voxel transformer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 13476–13487.
[59] Q. Xia et al., "3-D HANet: A flexible 3-D heatmap auxiliary network for object detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 61, pp. 1–13, 2023, Art. no. 5701113, doi: 10.1109/TGRS.2023.3250229.
[60] C. Zhou, Y. Zhang, J. Chen, and D. Huang, "OcTr: Octree-based transformer for 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 5166–5175.
[61] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2012, pp. 3354–3361.
[62] P. Sun et al., "Scalability in perception for autonomous driving: Waymo open dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2443–2451.
[63] H. Zhu et al., "VPFNet: Improving 3D object detection with virtual point based LiDAR and stereo data fusion," *IEEE Trans. Multimedia*, vol. 25, pp. 5291–5304, 2023, doi: 10.1109/TMM.2022.3189778.
[64] H. Yang et al., "Graph R-CNN: Towards accurate 3D object detection with semantic-decorated local graph," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2022, pp. 662–679.
[65] OpenPCDet Development Team. (2020) *OpenPCDet: An Open-Source Toolbox for 3D Object Detection From Point Clouds*. [Online]. Available: https://github.com/open-mmlab/OpenPCDet

Hiep Anh Hoang received the B.Eng. degree in control engineering and automation from the Hanoi University of Science and Technology, Hanoi, Vietnam, in 2020. He is currently pursuing the master's degree with Soongsil University, Seoul, South Korea.

His research interests include deep learning, 3-D perception of autonomous driving, and LiDAR point clouds.

Duy Cuong Bui received the B.S. degree in automatic control from the Hanoi University of Science and Technology, Hanoi, Vietnam, in 2020. He is currently pursuing the master's degree with Soongsil University, Seoul, South Korea.

His research interests include computer vision.

Myungsik Yoo received the B.S. and M.S. degrees in electrical engineering from Korea University, Seoul, South Korea, in 1989 and 1991, respectively, and the Ph.D. degree in electrical engineering from The State University of New York at Buffalo, Amherst, NY, USA, in 2000.

He was a Senior Research Engineer with the Nokia Research Center, Burlington, MA, USA. He is currently a Full-Time Professor with the School of Electronic Engineering, Soongsil University, Seoul. His research interests include visible-light communications, cloud computing, and machine learning.