Range-Aware Attention Network for LiDAR-Based 3D Object Detection With Auxiliary Point Density Level Estimation

Yantao Lu, Member, IEEE, Xuetao Hao, Yilan Li, Weiheng Chai, Shiqi Sun, and Senem Velipasalar, Senior Member, IEEE

Abstract—3D object detection from LiDAR data for autonomous driving has been making remarkable strides in recent years. Among the state-of-the-art methodologies, encoding point clouds into a bird’s eye view (BEV) has been demonstrated to be both effective and efficient. Different from perspective views, BEV preserves rich spatial and distance information between objects. Yet, while farther objects of the same type do not appear smaller in the BEV, they contain sparser point cloud features. This fact weakens BEV feature extraction using shared-weight convolutional neural networks (CNNs). In order to address this challenge, we propose Range-Aware Attention Network (RAANet), which extracts effective BEV features and generates superior 3D object detection outputs. The range-aware attention (RAA) convolutions significantly improve feature extraction for near as well as far objects. Moreover, we propose a novel auxiliary loss for point density estimation to further enhance the detection accuracy of RAANet for occluded objects. It is worth to note that our proposed RAA convolution is lightweight and compatible to be integrated into any CNN architecture used for detection from a BEV. Extensive experiments on the nuScenes and KITTI datasets demonstrate that our proposed approach outperforms the state-of-the-art methods for LiDAR-based 3D object detection, with real-time inference speed of 16 Hz for the full version and 22 Hz for the lite version tested on nuScenes lidar frames.

Index Terms—Object Detection, LiDAR Point Cloud, Bird’s Eye View, Autonomous Driving, Attention Convolution.

I. INTRODUCTION

With the rapid improvement of processing units, and thanks to the success of deep neural networks, the perception of autonomous driving has been flourishing in recent years. 3D object detection from LiDAR sensors is one of the important capabilities for autonomous driving. Early works employ 3D convolutional neural networks (CNNs), which have slow processing speeds and large memory requirements. In order to decrease the memory requirements and provide real-time processing, recent methodologies leverage voxelization and bird’s eye view (BEV) projection. Voxelization is widely implemented as a preprocessing method for 3D point clouds because of the computing efficiency, provided by more structured data, and performance accuracy [20], [40], [41], [51], [52]. In general, voxelization divides a point cloud into an evenly spaced grid of voxels, and then assigns 3D LiDAR points to their respective voxels. The output space preserves the Euclidean distance between objects and avoids overlapping of bounding boxes. This fact keeps object size variation in a relatively small range, regardless of their distance from the LiDAR, which benefits the shape regression during training. However, as shown in Fig. 1, voxels that are farther away from the ego-vehicle contain significantly fewer LiDAR points than the near ones. This leads to a situation where different representations may be extracted for an object at
different distances to the ego-vehicle. In contrast to perspective view, where feature elements are location-agnostic, BEV feature maps are location sensitive. Thus, different convolutional kernel weights should be applied to different locations of the feature map. In other words, location information should be introduced to the feature maps, and the convolutional kernels should be adjustable to the location information of corresponding feature maps. In this paper, we propose the Range-Aware Attention Network (RAANet), which contains the novel Range-Aware Attention Convolutional layer (RAAConv) designed for object detection from LiDAR BEV. RAAConv is composed of two independent convolutional branches and attention maps, which are sensitive to the location information of the input feature map. Our approach is inspired by the properties of BEV images, which are illustrated in Fig. 1. Points get sparser as the distance between an object and ego-vehicle increases. Ideally, for BEV feature maps, elements at different locations should be processed by different convolution kernels. However, applying different kernels will significantly increase the computational expense. In order to utilize the location information during BEV feature extraction, while avoiding heavy computation, we regard a BEV feature map as a composition of sparse features and dense features. We apply two different convolution kernels to simultaneously extract sparse and dense features. Each extracted feature map has half the channel size of the final output. Meanwhile, range and position encodings are generated based on the input shape. Then, each range-aware attention heatmap is computed from the corresponding feature map and the range and position encodings. Finally, the attention heatmaps are applied on the feature maps to enhance feature representation. The feature maps generated from two branches are concatenated channel-wise as the RAAConv output. Details are presented in Section III-B.

In addition, effects of occlusion cannot be ignored, since the same object may have different point distributions under different amounts of occlusion. Thus, we propose an efficient auxiliary branch, referred to as the Auxiliary Density Level Estimation Module (ADLE), allowing RAANet to take occlusion into consideration. Since annotating various occlusions is a time-consuming and expensive task, we design the ADLE to estimate a point density level for each object. If there is no occlusion, point density levels for near objects are higher than those of far objects. However, if a near object is occluded, its point density level decreases. Therefore, by combining range information and density level information, we are able to estimate existence of the occlusion information. ADLE is only used in the training stage for providing density information guidance, and can be removed in inference state for computational efficiency.

Contributions: The main contributions of this work include the following:

- We propose the RAAConv layer, which allows LiDAR-based detectors to extract more representative BEV features. In addition, RAAConv layer can be integrated into any CNN architecture used for LiDAR BEV.
- We propose a novel auxiliary loss for point density estimation to help the main network learn occlusion related features. This proposed density level estimator further enhances the detection accuracy of RAANet on occluded objects as well.
- We propose the Range-Aware Attention Network (RAANet), which integrates the aforementioned RAA and ADLE modules. RAANet is further optimized by generating anisotropic Gaussian heatmap based on the ground truth, which is introduced in Section III-D.

II. RELATED WORK

Majority of the LiDAR-based 3D object detection works can be categorized into two main groups: detection with anchors and without anchors. Additionally, there exist works [20], [43], [52] that encode point cloud data in the early stage, but they are out of scope for LiDAR-based 3D object detection network refactoring.

A. LiDAR-Based 3D Object Detection With Anchors

The fixed-shaped anchor regression approach [13], [14], [30] has been proposed such that intermediate features can be extracted. Girshick et al. [14] perform independent forward propagation on each region proposal. Region proposal task and 3D object detection task are compounded in [13], which avoids heavy convolution operation overhead and preserves the precision throughout the network. To decrease the computation further, Ren et al. [30] introduce a separate network to predict region proposals. This removes the time-consuming selective search algorithm and lets the network learn the region proposals. These works employ the two-stage detection scheme. However, the region-proposal-joint process (e.g. RPN) is still slow and memory intensive, and thus not compatible with real-time detection. To reduce computation as well as memory footprint during both training and inference phases, one stage detectors are exploited in many works [22], [23], [29], which can train object classification and bounding box regression directly, without pre-generated region proposals during training, and maintain a lower memory footprint at the same time. Redmon et al. [29] redefine LiDAR-based 3D object detection as a single regression problem, which employs an end-to-end neural network to do a single forward propagation to detect objects. Liu et al. [23] develop a multi-resolution-anchor technique to detect objects of a mixture of a scale as well as learn the offset to a certain extent than learning the anchor. Lin et al. [22] present a focal loss to tackle the dense and small LiDAR object detection problem while dealing with class imbalance and inconsistencies. Zhou and Tuzel [52] and Lang et al. [20] propose neural networks for point clouds, which open up new possibilities for 3D detection tasks. It should be noted that, although the aforementioned studies are real-time detection works, they actually involve the computation of all anchors’ regression corresponding for different anchor scales.

B. LiDAR-Based 3D Object Detection Without Anchors

To address the computation overhead and hyper-parameter redundancy introduced by anchor regression, and effectively process point cloud encodings, anchor-free LiDAR-based 3D
Fig. 2. **Overview of the proposed network.** The RPN-Points [20] is designed to extract the shared features from the BEV feature maps. The main detection heads and an auxiliary density estimation head are appended. The auxiliary head only exists during training and is omitted during inference. Finally, 3D bounding boxes are obtained from the main task. Our proposed RAAConv layer, illustrated in the lower part of the figure, replaces the original Conv2D layer. This layer contains two branches, and each branch applies position and range encodings to generate its range-aware attention heatmap. Then, extracted features in each branch are enhanced by the heatmap with a residual architecture. Those enhanced features are concatenated for the final output features.

Object detection has been exploited in many works [11], [16], [21], [35], [47], [50], [54]. Anchor-free LiDAR-based 3D object detection can be classified into two broad categories, namely center-based and key point-based approaches.

1) Center-Based Approach: In this approach [46], [49], the center points of an object are used to define positive and negative samples, instead of IoU [16], [35], [54]. This method significantly reduces the computation cost by predicting four distances from positive samples to boundary of objects to generate a bounding box. Tian et al. [35] predict one point at each position instead of defining several bounding boxes with different scales. Huang et al. [16] propose the DenseBox, a unified fully convolutional neural network, which simultaneously detects multiple objects with confidence scores. This method is primarily applied to face detection and cars detection. Zhu et al. [54] propose to attach online feature selection to each feature pyramid during training by setting up supervision signals through anchor-free branches. This work is demonstrated with feature pyramids of single-shot detectors.

2) Keypoint-Based Approach: In this approach [11], [21], [50], the key points are located by several predefined methods or self-learned models, and then generate the bounding boxes to classify the objects. Law and Deng [21] propose corner pooling, which aggregates the features of objects on the corner and groups corners with associative embeddings. Zhou et al. [50] develop a framework to predict four corner coordinates and center points simultaneously. However, the grouping algorithm and key points selection in these works introduce redundant false positive samples and effect the performance of detectors. To address this problem, Duan et al. [11] use both center points and corner points, which directly replace associative embedding with centripetal shifts. Although these paradigms are mainly proposed for optimizing data annotations, we have found that it also plays an essential role in feature representation and detection accuracy. In order to extract representative features, we focus on two major components: range-aware feature extraction and occlusion supervision.

**III. RANGE-AWARE ATTENTION NETWORK**

A. **Overview of the RAANet**

The main architecture of our proposed Range-Aware Attention Network (RAANet) is presented in Fig. 2. We incorporate ideas from CenterNet [46], [49] to build an anchor-free detector, and also introduce two novel modules: the Range-Aware Attention Convolutional Layer (RAAC) and the Auxiliary Density Level Estimation Module (ADLE).

RAANet takes the 3D LiDAR points as input and generates a set of 3D oriented bounding boxes as the output. Inspired by VoxelNet [52], we implement a feature extractor with 3D sparse convolutional network on the voxelized point clouds. This resulting features are reshaped and formed into BEV feature maps, which can be regarded as a multi-channel 2D image. The Region Proposal Network (RPN) takes this BEV feature maps as
input and employs multiple down-sample and up-sample blocks to generate a high-dimensional feature map. In addition to the detection heads in main task, we propose an auxiliary task for point density level estimation to achieve better detection performance. Meanwhile, the RAACConv layers are utilized in all convolutional modules. The details of the RAACConv layer are shown in the bottom half of Fig. 2. Center probability heatmap is generated from the detector head, and maximums in local neighborhoods are treated as the centers of ground-truth bounding boxes (bbox). The regression of bbox attributes, including offsets, dimensions and orientations, are computed from the other heads in the main detection task. Parallel to the main task, the proposed ADLE, as an auxiliary task, estimates the point density level for each bounding box. In general, ADLE classifies each bounding box into different density levels, based on the number of points in each bbox. ADLE is a portable block for the whole network, i.e. it is employed only during training and is not used during inference for higher computational efficiency.

For the training phase, the loss components, namely the heatmap classification loss, bbox regression loss and density-level auxiliary loss are denoted as $l_{hm}$, $l_{box}$ and $l_{aux}$, respectively. The total loss function of RAANet is formulated as

$$l = l_{hm} + \lambda_{box} l_{box} + \lambda_{aux} l_{aux}, \quad (1)$$

where $\lambda_{box}$ and $\lambda_{aux}$ are scalar weights for balancing the multi-task loss. $l_{hm}$ and $l_{aux}$ are designed as penalty-reduced focal losses [22], [49], and $l_{box}$ is designed as the smooth L1 loss.

B. Range-Aware Attentional Convolutional Layer

We propose the Range-Aware Attentional Convolutional Layer (RAACConv), which, by leveraging the specially designed attention blocks, becomes sensitive to range and position information. This is the major difference between the proposed RAACConv and a traditional convolutional layer. RAACConv is employed in the aforementioned RPN and detection heads. As shown in Fig. 2, RAACConv first utilizes two sets of convolutional kernels to extract an intermediate feature map for each branch. Then, the position encodings are generated and embedded into intermediate feature maps. Two range-aware heatmaps are calculated by a series of convolution and pooling operations using the intermediate feature maps and range encodings. More specifically, given an input feature map $I$ $\in$ $\mathbb{R}^{H_{in} \times W_{in} \times C_{in}}$, two separate intermediate feature maps, $F_{a}$, $F_{b}$ $\in$ $\mathbb{R}^{H_{out} \times W_{out} \times C_{out}}$, are generated by

$$F_{i} = \zeta(I, w_{i}^{f}), \quad i = a, b, \quad (2)$$

where $\zeta(\cdot)$ represents the convolution operation and $w_{i}^{f}$ denotes the corresponding convolution kernel.

Meanwhile, the position encoding map $\xi$ $\in$ $\mathbb{R}^{H_{out} \times W_{out} \times 2}$ and range encoding map $\rho$ $\in$ $\mathbb{R}^{H_{out} \times W_{out} \times 1}$ are generated from the intermediate feature maps $F_{a}$, $\xi$ and $\rho$ are calculated from shape information and do not depend on the specific values inside the tensor $F_{a}$. There are two components of $\xi$: row encoding $r$ and column encoding $c$. $\xi$ and $\rho$ are generated as follows:

$$r_{ij} = 2\lfloor i - \frac{H}{2} \rfloor / H, \quad c_{ij} = 2\lfloor j - \frac{W}{2} \rfloor / W$$

Values of $r$, $c$ are bounded in $[0, 1]$. Values of $\rho$ are bounded in $[-1, 1]$.

Since the proposed network is designed by using two different convolution kernels, to extract dense and sparse features separately, we append $\xi$ to intermediate feature map $F_{a}$ channel-wise, and reversely append $1 - \xi$ to $F_{b}$. The generated features are denoted by $F_{a}$ and $F_{b}$. Maximum pooling, mean pooling and $1 \times 1$ conv2D are applied on $F_{a}$ and $F_{b}$ separately to obtain the spatial embeddings $F_{a}'$ and $F_{b}'$. Then, similar to positional appending, the range encoding map $\rho$ is appended to $F_{a}'$, and $-\rho$ is appended to $F_{b}'$. The appended features are processed by a $3 \times 3$ conv2D layer followed by sigmoid activation to obtain the range-aware attention heatmaps $f_{a}$ and $f_{b}$. The heatmaps $f_{a}$ and $f_{b}$ are then multiplied by learnable scalars $\gamma_{a}$ and $\gamma_{b}$, respectively. $\gamma_{a}$ and $\gamma_{b}$ are initialized as 1.0 and their values are gradually learned during training [48]. The output feature maps $F_{a}''$ and $F_{b}''$ are calculated from $F_{a}'$, $f_{a}$, $\gamma_{a}$, and $F_{b}'$, $f_{b}$, $\gamma_{b}$ using a residual connection, which is defined as:

$$F_{i}'' = (\gamma_i f_i) \otimes F_{i} + F_{i}^{aux}, \quad i = a, b \quad (4)$$

where $\otimes$ is the operation performing channel-wise stacking and element-wise multiplication. The final output for RAACConv is the channel-wise concatenation of $F_{a}''$ and $F_{b}''$, which has a size of $H_{out} \times W_{out} \times C_{out}$.

It is worth noting that the proposed RAACConv can be readily plugged in any convolutional network for LiDAR BEV detection.

C. Auxiliary Density Level Estimation Module (ADLE)

The Auxiliary Density Level Estimation Module (ADLE) is an additional classification head that is parallel to the main task heads. The ADLE module indicates the point density level for each bbox. In general, we design the ADLE module to estimate the number of LiDAR points inside a detected bbox. We have empirically found out that this estimation is straightforward yet beneficial for the detector. Moreover, we divide the number of points into three density bins and let the ADLE perform a classification task, instead of a value regression task, which helps training to converge better. More specifically, the three bins are labeled as sparse (label 1), adequate (label 2) and dense (label 3). We have statistically analyzed the distribution of the number of points and employed two thresholds $T_{0}^{i}$, $T_{1}^{i}$ to determine the density level for each object class $i$. The three density levels are defined as:

$$D_{i}(n) = \begin{cases} 
1 & \text{if } n < T_{0}^{i} \\
2 & \text{if } T_{0}^{i} \leq n < T_{1}^{i} \\
3 & \text{if } n \geq T_{1}^{i}
\end{cases} \quad (5)$$

where $n$ denotes the number of points, and $D_{i}$ represents the density level for the $i$-th class. The implementation details of $T_{0}^{i}$, $T_{1}^{i}$ are shown in Section IV. As shown in Fig. 3, the unoccluded instances near the ego-vehicle have high point density levels and faraway instances have low density levels. However,
when an instance is occluded, the density level will be lower than the unoccluded case at the same distance. Since we incorporate position and range information into RAACConv, ADLE is able to further help RAANet extract occlusion information by supervising the point density levels.

D. Anisotropic Gaussian Mask

Anisotropic Gaussian Mask represents the centerness probability for each position in an annotated bbox. The centerness heatmap is generated as the ground truth for the classification head. In this work, we propose an Anisotropic Gaussian Mask, shown in Fig. 3, to generate a 2D Gaussian distribution, to be used for the centerness probabilities of a given oriented bbox. More specifically, the Gaussian distribution \( N(\mu, \Sigma) \) is designed in an anisotropic way, i.e. the \( \sigma \) values for the 2 dimensions, which are the diagonals of \( \Sigma \), and denoted as \( \sigma_0, \sigma_1 \), are different, and their values are determined from the length and width of the bbox, respectively. The procedure for generating the center heatmap, denoted by \( M \), can be formulated as:

\[
M_{C_H, C_W, H, W, s}(x, y) = \mathbb{I}(N(\mu, \Sigma)(x, y))
\]  

where,

\[
\mathbb{I} = \left\{ x \in \left[ C_W - \frac{W}{2}, C_W + \frac{W}{2} \right], \ y \in \left[ C_H - \frac{H}{2}, C_H + \frac{H}{2} \right] \right\}
\]

\[
\mu = [C_W, C_H]^T
\]

\[
\Sigma = \begin{bmatrix}
W/d_{cls} & 0 \\
0 & 0H/d_{cls}
\end{bmatrix}
\]

and \((C_H, C_W)\) and \((H, W)\) denote the center location and dimensions of a bbox, respectively, \(s\) indicates the ground truth class for the bbox, \( \mathbb{I}(\cdot) \) is the indicator function and \( \mathbb{I} \) denotes the set of locations that are inside the bbox. \( d_{cls} \) is a decay factor determining the sharpness of generated heatmap. The value of \( d_{cls} \) depends on the class of a given bbox.

E. Loss Functions

Following (1), the center heatmap loss \( L_{hm} \) is defined as:

\[
L_{hm} = -\frac{1}{N} \sum_{\hat{p}} \{(1 - h)^{\alpha} \log(h)I_{h > 1 - \epsilon} + (1 - \hat{h})^{\beta} \log(1 - h)I_{\hat{h} < 1 - \epsilon}\},
\]

where \( h \) and \( \hat{h} \) are the ground-truth and predicted heatmap values, respectively. \( \alpha \) is a small value for numerical stability. \( \alpha \) and \( \beta \) are the parameters for controlling the penalty-reduced focal loss. For all the experiments, we follow the setting in [34], i.e. \( \epsilon = 1e - 3 \), \( \alpha = 2 \) and \( \beta = 4 \).

The bounding box head is responsible for the location, dimension and orientation regression, which are represented by the following parameters: \( \Gamma = [\delta_x, \delta_y, \delta_z, l, w, h, \theta] \), where \( \delta_x, \delta_y, \delta_z \) are the box center offsets with respect to the corresponding voxel centers; and \( l, w, h \) and \( \theta \) denote the length, width, height and yaw of a bounding box. We apply smoothed L1 (SL1) losses to regress \( \sin(\theta) \) and \( \cos(\theta) \) for the \( \theta \) attribute, and directly regress actual values for other attributes:

\[
L_{\theta_i} = SL1(\sin(\hat{\theta}_i) - \sin(\hat{\theta}_i)) + SL1(\cos(\hat{\theta}_i) - \cos(\hat{\theta}_i))
\]

\[
L_{\Gamma_{1:3}} = SL1(\Gamma_{1:3} - \hat{\Gamma}_{1:3})
\]

where, \( \hat{\cdot} \) denotes predicted values corresponding to original symbols. The total loss for box regression is:

\[
L_{box} = \frac{1}{N} \sum_{i} (L_{\theta_i} + L_{\Gamma_{1:3}}),
\]

where \( N \) is the number of bbox samples and \( \mathbb{I} \) equals 1 if the ground truth heatmap value of the \( i \)th feature map pixel is larger than a threshold \( \tau \), which is set to 0.2 in the experiments.

IV. EXPERIMENTS

We introduce the implementation details of the proposed RAANet and evaluate its performance in multiple experiments. The inference results are uploaded to and evaluated by the nuScenes official benchmark [25]. The diagnostic analysis of each component is shown in the ablation studies in Section IV-E.

A. Datasets

We perform the evaluations on the nuScenes [2] and KITTI [12] datasets, which are publicly available, large-scale and challenging datasets for autonomous driving. The full nuScenes dataset includes approximately 390K LiDAR sweeps and 1.4M object bounding boxes in 40 K key frames. The dataset contains 1000 scenes in total, of which 700 are separated for training, 150 for validation and 150 for testing. There are a total of 10 classes: \{car, truck, bus, trailer, construction vehicle, traffic cone, barrier, motorcycle, bicycle, pedestrian\} . The scenes of 20 sec. length are manually selected to cover a diverse and interesting set of driving maneuvers, traffic situations and unexpected behaviors. The dataset was captured with a 32-channel spinning LiDAR sensor with 360° horizontal and −30 to +10° vertical
FOV. The sensor captures points, which are within 70 meters with ±2 cm accuracy and returns up to 1.39M points per second.

We also use the KITTI dataset to benchmark performance. The 3D object detection dataset consists of 7481 training and 7518 testing samples for the point clouds, including a total of 80256 labeled objects. Following the experimental setting used in [20], we divide the official training samples into 3712 training samples and 3769 validation samples. Meanwhile, considering the test submission requirement, we create 784 samples as a minival validation set, and use the remaining 6697 samples to train. Since the KITTI benchmark only provides the ground truth annotations of objects that are visible in camera images, we followed the standard literature practice on KITTI of only using LiDAR points that project into the image, and trained one network for cars to benchmark our model performance.

B. Implementation Details

RAANet is implemented in PyTorch framework [26] with an anchor-free object detection architecture. The weights $\lambda_{\text{box}}$ and $\lambda_{\text{center}}$ in loss function (1) are set to 0.25 and 0.2, respectively. The decay factor $d$ in Section III-D is set to 3 for large size classes (car, truck, bus, trailer, construction_vehicle), to 6 for small object classes (traffic_cone, barrier, motorcycle, bicycle, pedestrian). As for the parameters in Section III-E, we set $\epsilon$, $\alpha$, $\beta$ and $\tau$ to 0.001, 2.0, 4.0 and 0.2, respectively.

For nuScenes, all classes are trained end-to-end. The resolution for voxelization is set to $[0.075 m, 0.075 m, 0.2 m]$ and the voxelization region is $[-54 m, 54 m] \times [-54 m, 54 m] \times [-5 m, 3 m]$. The model is trained by 20 epochs. Initial learning rate is set to $1e - 3$ and then is modified by one-cycle adjuster [33]. Adam optimizer [17] with decay parameters 0.9, 0.99 is applied. The models are trained by 8 Nvidia V100 GPUs, and the batch size is set to 4 for each GPU.

For KITTI experiments, the resolution for voxelization is set to $[0.16 m, 0.16 m, 0.5 m]$ and the voxelization region is $[0 m, 69.12 m] \times [-39.69 m, 39.68 m] \times [-3.5 m, 1.5 m]$. The model is trained by 80 epochs. Initial learning rate is set to $2e - 4$ and is decayed by 0.8 for every 15 epochs. Adam optimizer with decay parameters 0.9, 0.99 is applied.

We determine the threshold values of ADLE by conducting an empirical analysis of the relationship between the number of LiDAR points reflected from each labeled object and the distance from the ego car. We first categorize all objects into two classes based on size: “large” and “small”. For the KITTI dataset, “large” objects include cars, whereas “small” objects encompass pedestrians and cyclists. For the nuScenes dataset, the “large” category comprises cars, trucks, buses, trailers, and construction vehicles, and the “small” category includes pedestrians, cyclists, bicycles, traffic cones, and barriers. The statistical information is shown in Fig. 4. In each plot, we represent the statistical distribution of point counts at varying distances for these object categories. The box plots illustrate the quartiles and median values, which provide a robust summary of the distribution that is less susceptible to outliers. The “+” marks indicate the mean number of points, which we found to be a reliable indicator of the typical point density that can be expected for objects at different ranges. Based on our empirical experiments and evaluations, we set the thresholds at the mean values obtained at two specific ranges: 20 meters and 40 meters. The specific threshold values were set as follows: $T_0 = 189$, $T_1 = 347$ for nuScenes large objects; $T_0 = 24$, $T_1 = 74$ for nuScenes small objects; $T_0 = 96$, $T_1 = 640$ for KITTI large objects; $T_0 = 55$, $T_1 = 263$ for KITTI small objects. These values were selected because they represent a balance between detection reliability and computational efficiency.

For future work, we plan to develop a more generalized thresholding approach by incorporating relevant hyperparameters as part of the input to our Auxiliary Density Level Estimation (ADLE) Module. These hyperparameters will include factors such as the number of LiDAR sensor beams, beam distribution patterns, and perception ranges, among others. By integrating these sensor-specific characteristics, we aim to create a model that can dynamically adjust its thresholds based on the properties of the sensor data it processes. This approach will not only streamline the adaptation process across different LiDAR systems but also improve the robustness and flexibility of our method when applied to a variety of sensor configurations.

Two versions of the RAANet are designed in this work: i) the full version RAANet applies RAACConv layers throughout all modules; ii) RAANet-lite version applies RAACConv layers only at the detection heads. Both designs outperform SOTA baselines. Compared to the RAANet, RAANet-lite has faster inference speed, with a small drop in performance in terms of the mAP and NDS metric.

C. Training and Inference

Several data augmentation strategies are implemented in the training phase. They are random flipping along the x-axis, random scaling for the point clouds, and global rotation around the z-axis with a random angle sampled from $[-\pi/4, \pi/4]$. Other details follow original the CenterPoint work, Class-Balanced Grouping and Sampling (CBGS) was applied during our
experimental setup to address class imbalance issues. The sampling is utilized to guarantee enough objects are appeared, yet grouping is not utilized during training. We utilized Ground Truth (GT) sampling as part of our data augmentation strategy. However, we did not incorporate the fade strategy from Point Augmenting in our experiments. Point shuffle, rotation, scaling, translation are applied as point augmentation. In addition, the objects that has fewer points that thresholds are filtered during sampling.

During inference, we reuse the checkpoint with the best mean average precision (mAP) value and evaluate on the official lidar-only nuScenes and KITTI evaluation metrics. The detection score threshold is set to 0.1 and IoU threshold is set to 0.2 for non-maximal suppression.

The inference speed is tested on the nuScenes with 2x Intel Xeon 5218R CPU and 1x Nvidia V100 GPU. The reason is that the nuScenes data is much closer to the real autonomous driving scenarios, whereas KITTI dataset only stores the front views of the scenes. Under GPU inference environment, the latency for the full version of RAANet and RAANet-lite are measured as 62 ms per frame (16.13 Hz FPS) and 45 ms per frame (22.22 Hz FPS), respectively. Both versions satisfy the real-time 3D detection criteria. Under CPU only environment, the latency for the full version of RAANet and RAANet-lite are measured as 2998.14 seconds per frame and 2182.41 ms per frame, respectively.

D. Results

1) Results on the NuScenes Dataset: We present the results obtained on the test set with both our lite-version and full-version models, shown in Tables I and II, in terms of different metrics. The architecture of the full version is illustrated in Fig. 2. All detection results are evaluated in terms of mean average precision (mAP), mean Average Scale Error (mASE), mean Average Orientation Error (mAOE), Average Velocity Error (AVE), Average Attribute Error (AAE) and nuScenes detection score (NDS). The NDS is weighted summed from mAP, mATE, mAVE, mAASE and mAAME. It firstly converts the errors to scores by \( \max(1 - \text{error}, 0.0) \). Then a weight of 5.0 is assigned to mAP, and a weight of 1.0 is assigned to the other five scores. We compare our work with 16 other SOTA models that have open-source codes. As can be seen in Table I, our proposed RAANet outperforms all the other models in terms of mAP, and the RAANet-lite takes the second place. For individual classes, RAANet achieves the best AP for 9 out of 10 classes, and RAANet-lite takes the second place for 7 out of 10 classes. As shown in Table II, the RAANet provides the best performance in terms of mAP, mATE, mAASE and mAAME metrics, and the 3rd best performance in terms of the mAVE and mAAME metrics. The mASE of the RAANet only shows a small gap compared with the top performances on the benchmark. The RAANet-lite takes the second best place for NDS and mAP metrics.

2) Results on the KITTI Dataset: We present the results, obtained on the test set with our full-version models in Table III. All detection results are evaluated by the average precision (AP) metric for BEV car object detection. We compare our work with other state-of-the-art models that leverage LiDAR points data. As can be seen in Table III, our proposed RAANet outperforms all the other models in terms of mAP, and the RAANet-lite takes the second place. For individual classes, RAANet achieves the best AP for 9 out of 10 classes, and RAANet-lite takes the second place for 7 out of 10 classes. As shown in Table III, our proposed RAANet outperforms all the other models in terms of the average APs for “Easy”, “Moderate” and “Hard” testing samples, which are defined in the KITTI dataset. Thus, when all these testing samples are considered together, RAANet achieves the best detection performance.

E. Ablation Studies

In this section, we provide the results of the ablation studies for different components and critical parameters of our approach, more specifically the attention module, decay factor \( d \) and
TABLE II
COMPARISON WITH THE STATE-OF-THE-ART MODELS FOR 3D DETECTION ON THE NUSCENES TEST DATASET FOR DIFFERENT PERFORMANCE METRICS

| Method         | mAP↑ | mATE↓ | mASE↓ | mAOE↓ | mAVE↓ | mAAE↓ | NDS↑ |
|---------------|------|-------|-------|-------|-------|-------|------|
| PointPillars  | 0.401| 0.392 | 0.269 | 0.476 | 0.270 | 0.102 | 0.550|
| SA-Det3D      | 0.470| 0.317 | 0.248 | 0.438 | 0.300 | 0.129 | 0.592|
| InfoFocus     | 0.395| 0.363 | 0.265 | 1.132 | 1.000 | 0.395 | 0.395|
| CyInNetRG     | 0.385| 0.272 | 0.243 | 0.383 | 0.293 | 0.126 | 0.661|
| SARFNET       | 0.324| 0.400 | 0.249 | 0.763 | 0.272 | 0.090 | 0.484|
| PanoNet3D     | 0.545| 0.298 | 0.247 | 0.395 | 0.338 | 0.136 | 0.631|
| PolarStream   | 0.538| 0.339 | 0.264 | 0.491 | 0.339 | 0.123 | 0.613|
| MEGVII        | 0.528| 0.300 | 0.247 | 0.379 | 0.245 | 0.140 | 0.633|
| PointRCNN     | 0.510| 0.301 | 0.245 | 0.378 | 0.326 | 0.138 | 0.616|
| SSN-v2        | 0.506| 0.339 | 0.245 | 0.429 | 0.266 | 0.087 | 0.616|
| ReconfigPP-v2 | 0.325| 0.388 | 0.250 | 0.486 | 0.306 | 0.137 | 0.506|
| ReconfigPP-v3 | 0.485| 0.328 | 0.244 | 0.439 | 0.274 | 0.244 | 0.590|
| HotSpotNet    | 0.593| 0.274 | 0.239 | 0.384 | 0.333 | 0.133 | 0.660|
| MMDec3D       | 0.575| 0.316 | 0.256 | 0.409 | 0.236 | 0.124 | 0.653|
| CenterPoint   | 0.603| 0.262 | 0.239 | 0.361 | 0.288 | 0.136 | 0.673|
| CViNCv-ens    | 0.582| 0.284 | 0.241 | 0.372 | 0.224 | 0.126 | 0.666|
| RAA-Net-lite  | 0.620| 0.277 | 0.244 | 0.365 | 0.212 | 0.126 | 0.687|
| RAA-Net (Ours)| 0.635| 0.259 | 0.240 | 0.326 | 0.210 | 0.125 | 0.702|

The best results are marked in bold and second best results are marked in blue color.

TABLE III
COMPARISON WITH THE STATE-OF-THE-ART MODELS FOR BEV DETECTION ON THE KITTI TEST DATASET

| Method            | BEV (AP↑) | Easy | Moderate | Hard | AVG |
|-------------------|-----------|------|----------|------|-----|
| AVOD[19]          | 0.898     | 0.830| 0.783    | 0.843|     |
| MV3D[6]           | 0.865     | 0.790| 0.722    | 0.792|     |
| PointRCNN[32]     | 0.921     | 0.874| 0.827    | 0.874|     |
| Fast PointRCNN[9] | 0.909     | 0.878| 0.805    | 0.864|     |
| P-FPointNet[28]   | 0.912     | 0.847| 0.748    | 0.835|     |
| F-ConvNet[40]     | 0.915     | 0.858| 0.761    | 0.845|     |
| 3DSSD[45]         | 0.927     | 0.890| 0.859    | 0.892|     |
| VoxelNet[53]      | 0.880     | 0.784| 0.713    | 0.792|     |
| SECOND[43]        | 0.894     | 0.838| 0.786    | 0.839|     |
| PointPillars[20]  | 0.901     | 0.866| 0.828    | 0.865|     |
| CenterPoint-pcdet[47] | 0.885     | 0.851| 0.812    | 0.849|     |
| RAA-Net (Ours)    | 0.953     | 0.894| 0.843    | 0.897|     |

We list Car AP scores for easy, moderate and hard samples. The best results are marked in bold.

(for the anisotropic Gaussian mask) and $\lambda_{aux}$ (weight for the density level estimation loss). The models are evaluated on the nuScenes validation set, since it has much more samples than the KITTI. mAP is calculated for comparison. The baseline is the CenterPoint [46] with isotropic Gaussian Mask and traditional convolutions.

The structures of the RAAConv layers employed in the ablation studies are shown in Fig. 5. In order to evaluate the effectiveness of our proposed RAAConv layer, we test each sub-component inside the RAAConv. Fig. 5 shows three main variants of RAAConv: a) Dual branch module built with the original convolution, where each branch has half the number of output channels; b) Single branch module built with range-aware attention, yet without the learnable scalar $\gamma$; c) Dual branch module built with all sub-components. The reverse encoding indicates different positional encodings on the two attention branches, in order to extract dense and sparse features separately. The last sub-figure (Fig. 5(d)) shows the structure of the RAA block that is used for generating attention heatmaps.

Fig. 5. RAAConv structures use in the ablation studies. (a) Dual branch module built with the original convolution, where each branch has half the number of output channels; (b) Single branch module built with range-aware attention, yet without the learnable scalar $\gamma$; (c) Dual branch module built with all sub-components; (d) Structure of the RAA block that is used to generate attention heatmaps.

Anisotropic Gaussian Mask: We evaluate the contribution of the Anisotropic Gaussian Mask by varying the decay factor $d$. The ablation study results presented in Fig. 6(a) show that, with anisotropic Gaussian mask, the performance improvement (in terms of AP) for the ‘Car’ class is more compared to the
TABLE IV
Ablation Study for the Range-Aware Attention Network

| Model                  | RAACConv Dual Branch | Attention γ | 2D Gaussian l_{aux} = 0.25 | ADLE | mAP (%) |
|------------------------|----------------------|-------------|-----------------------------|------|---------|
| Baseline               | -                    | -           | -                           | -    | 60.28   |
| + RAACConv             | ✓                    | ✗           | ✗                           | ✓    | 61.25   |
|                        | ✓                    | ✓           | ✓                           | ✗    | 61.49   |
| + 2D Gaussian          | ✓                    | ✓           | ✓                           | ✓    | 61.83   |
| + auxiliary loss       | ✓                    | ✓           | ✓                           | ✗    | 62.27   |

Various models are built by different combinations of three attributes: RAACConv, Anisotropic Gaussian Mask (2D Gaussian for short) and ADLE. For RAACConv, there are four sub-components: dual branch structure, range-aware attention heatmap and learnable scalar γ (Fig. 2).

Fig. 6. Ablation studies on Anisotropic Gaussian Mask and ADLE. (a) shows AP results with different Gaussian Σ decay factors, which is denoted as d in (7). (b) shows different ADLE loss weights denoted as λ_{aux} in (1).

‘Pedestrian’ class. One of the probable reasons is that the annotated bboxes for the pedestrian class have similar widths and lengths. This leads the anisotropic Gaussian mask to fall back to the original isotropic Gaussian mask. Additionally, we set d = 3 for large objects and d = 6 for the small objects in our main experiments. In general, IoU-based evaluation is more sensitive on small objects than large objects, and sharper Gaussian masks help achieve higher accuracy values for small objects.

Auxiliary Density Level Estimation Module: The AP results obtained with various weights of the loss for the auxiliary component (λ_{aux}), are shown in Fig. 6(b). The loss weight values are chosen between 0 and 1. Based on these results, We set λ_{aux} = 0.2 for the main experiments.

Range-Aware Attentional Convolution: Table IV shows the performance of the baseline as well as several types of the Range-Aware Attention Network. For RAACConv, there are three sub-components: dual branch structure, range-aware attention heatmap and learnable scalar γ (Fig. 2). Each of these components contributes to the improvement of detection performance. We observe that the combination of dual-branch structure and range-aware attention provides 0.9% gain on the small object. With the learnable scalar γ, the dual-branch attentions can be adaptive to the variation of the training data resulting in further improvement.

F. Generalizability

The proposed RAANet is further evaluated by a different backbone from PointPillar [20]. More specifically, we replace the original 3D voxel-based backbone with a 2D pillar-based backbone. The pillar size is set to (0.2, 0.2). The perception range is set to $[-51.2, 51.2] \times [-51.2, 51.2] \times [-5, 3]$. The baseline is trained with the CenterPoint-PointPillar version. As shown in Table V, our proposed RAANet still provides a competitive performance with each component contributing to the performance improvement.

In the aforementioned main experimental subsections, we chose to highlight the BEV results as supplementary information to the more extensive nuScenes experiments. We believe that the nuScenes dataset, being newer and more commonly used in recent research, is more indicative of our method’s performance in contemporary contexts. As a complementary information, in this part, we further show more comparison results of the proposed method and its baseline PointPillar on KITTI dataset. The results are shown in Table VI. It can be seen that aided by the proposed range-aware convolution and ADLE, the baseline model substantially improves across multiple metrics, including “Detection”, “Orientation”, “3D Detection”, and “Bird’s Eye View”. Specifically: For Car (Detection), we achieved an increase from 94.00% to 96.81% for easy, from 91.19% to 93.63% for moderate, and from 88.17% to 90.39% for hard difficulty levels. For car (Orientation), there was an improvement from 93.84% to 96.78% for easy, from 90.70% to 93.39% for moderate, and from 87.47% to 89.88% for hard levels. For Car (3D Detection), we observed an enhancement from 82.58% to 87.79% for easy, from 74.31% to 79.15% for moderate, and...
from 68.99% to 73.92% for hard levels. For Car (Bird’s Eye View), our results showed an increase from 90.07% to 95.25% for easy, from 86.56% to 89.41% for moderate, and from 82.81% to 84.34% for hard levels.

The intrinsic differences in LiDAR sensor specifications, including the number of beams and beam distribution, significantly impact point density and, by extension, the determination of appropriate thresholds for our ADLE approach. As noted, the KITTI dataset, captured with a 64-beam LiDAR sensor, generally contains a higher number of points per object compared to the nuScenes dataset captured with a 32-beam LiDAR sensor under similar conditions. Besides, the beam distribution of different sensors can be different. For instance, LiDAR systems with imbalanced beam distribution tend to have sparser lower beams compared to the denser middle beams. This phenomenon is evident in the KITTI dataset, where objects within 15 meters may exhibit a lower number of points compared to those at 20 and 25 meters (as shown in Fig. 4(c) and (d)). Such sensor-specific characteristics underscore the impracticality of applying universal threshold values across datasets. To address this, we performed separate analyses for each dataset, leading to individually tailored thresholds that reflect the unique point density profiles of the respective LiDAR systems. This approach is detailed in the previous subsection. Moreover, considering the industry practice where detectors trained on public datasets are often fine-tuned with proprietary data before deployment on real-world autonomous driving platforms, it is feasible to determine the ADLE thresholds for the self-collected data. Subsequently, the ADLE module can be incorporated into the fine-tuning pipeline, allowing for adjustments that account for the unique characteristics of the deployment-specific LiDAR sensor. This adaptable framework ensures that the ADLE method remains relevant and effective across a variety of LiDAR systems, facilitating its extension to new datasets and sensor technologies in the rapidly evolving field of autonomous driving.

### Table V

| Models       | RAACConv | 2D Gaussian | ADLE | mAP(%) |
|--------------|----------|-------------|------|--------|
| Baseline     | X        | X           | X    | 46.66  |
| RAANet       |          |             |      | 47.46  |

The baseline and comparison models are built with a PointPillar backbone and validated on nuScenes. Comparison models are composed of various combinations of RAACConv, 2D Gaussian, and ADLE.

### Table VI

| Benchmark        | Easy       | Moderate    | Hard        |
|------------------|------------|-------------|-------------|
| Car (Detection)  | 96.81% (94.00%) | 93.63% (91.19%) | 90.39% (88.17%) |
| Car (Orientation)| 96.78% (93.84%) | 93.39% (90.70%) | 89.88% (87.47%) |
| Car (3D Detection)| 87.79% (82.58%) | 79.15% (74.31%) | 73.92% (68.99%) |
| Car (Bird’s Eye View) | 95.25% (90.07%) | 89.41% (86.56%) | 84.34% (82.81%) |

### Table VII

| State-of-the-Art Models | mAP | NDS | mAP | NDS |
|-------------------------|-----|-----|-----|-----|
| SA-VFE                  | 59.9| 67.0| 60.6| 68.1|
| LargeKernel3D           | 63.3| 69.1| 63.9| 69.9|
| BEVF-L                  | 63.9| 69.6| 64.3| 70.1|
| PillarNet               | 66.0| 71.4| 66.2| 71.6|

In order to evaluate the generalizability of adapting the proposed RAANet on different base architectures, we perform comparison experiments on various SOTA networks. The evaluation is conducted on the nuScenes dataset, and the SOTA networks SA-VFE [18], LargeKernel3D [8], BEVF-L [24], and PillarNet [15] are selected as baselines. As shown in Table VII, the performance is enhanced for all three baselines when combined with the proposed RAANet methods.

### G. Qualitative Analysis

Pairs of example outputs, obtained by the baseline and the proposed RAANet are provided in Fig. 7 for qualitative comparison. The sample data is selected from the nuScenes validation dataset. In a pair, images on the left are the results obtained from the baseline, and the images on the right are the results from the proposed RAANet. It can be observed that RAANet increases true positives and suppresses false positives at far or occluded regions.

Fig. 7(a) Illustrates a complex urban intersection with multiple static and dynamic objects, including vehicles, pedestrians, and street signs. The red bounding boxes represent the ground truth for various objects within the scene, and the blue boxes are the predicted detections by RAANet. Fig. 7(b) Shows a straight roadway segment with various automobiles and roadside
Fig. 7. **Qualitative results.** (Please zoom-in for details). For each image pair, images on the left are the results inferred from the baseline, and the images on the right are the results from the proposed RAANet. Red and blue boxes denote the ground truth and predictions, respectively.

Fig. 8. **Detection and attention visualization.** For each sub-figure, the image on the left is the all-class detection result, inferred from our proposed RAANet on the nuScenes validation dataset, and the two images on the right are the attention heatmaps from two convolutional branches in the RAAConv layer. Different color distributions represent how much attentions is paid on the different ranges.

infrastructure. This figure highlights RAANet’s detection capabilities in a scenario with different object classes and movement patterns. Fig. 7(c) Depicts a more challenging scene with objects at various distances and occlusions, demonstrating RAANet’s robust detection in dense urban environments. Fig. 7(d) Presents a scenario similar to (c) but from a different angle, further emphasizing the model’s ability to maintain high detection performance despite viewpoint changes. In each left sub-figure, we aim to show the all-class detection capabilities of RAANet. The color-coded bounding boxes are used to visually compare
the ground truth and the predictions made by our network. The right two images in each sub-figure are attention heatmaps generated from the two convolutional branches within the RAANet layer. These heatmaps are color-coded to represent the varying levels of attention the model pays to different areas within the scene, which is crucial for understanding the model’s focus areas and decision-making process. The heatmaps provide an intuitive understanding of how RAANet processes a scene: areas with higher attention levels are likely to contain objects of interest, and the color distribution gives insights into how the model prioritizes different parts of the input data. This is particularly useful for analyzing the model’s behavior in complex scenes, where objects may be close to each other or partially occluded.

Fig. 9 shows more visualization examples on all-class detection of the RAANet as well as the attention heatmaps generated from RAANet. For each sub-figure, the image on the left is the all-class detection result obtained from our proposed RAANet, and the two images on the right are the attention maps from both convolutional branches in the RAANet layer. It can be observed that both attention modules represent the positional heatmaps based on the BEV. In general, the attention heatmap from branch ‘a’, highlights the regions with dense point clouds or the ego-vehicle vicinity. The attention heatmap from branch ‘b’, on the other hand, pays attention to the regions with sparse point clouds or are farther from the ego vehicle. The combination of both branches effectively extracts powerful BEV features and generates superior 3D object detection results.

Fig. 9 shows some hard examples that were misjudged by our proposed method. In Fig. 9(a), although our RAANet successfully generates the detection results for all ground truths, some of the detection headings are not regressed accurately enough due to the long distance and sparse points inside the objects. In Fig. 9(b), some false positives appear on severely occluded regions.

V. Future Works and Conclusion

In this paper, we have introduced the Range-Aware Attention Network (RAANet) to improve the performance of 3D object detection from LiDAR point clouds. The motivation behind the RAANet is that, in Bird’s Eye View (BEV) LiDAR images, objects appear very different at various distances to the ego-vehicle, and thus, there is a need to avoid using shared-weight convolutional feature extractors. In particular, we have leveraged position and range encodings to design a RAACConv layer with two independent convolution branches, which separately focus on sparse and dense feature extraction. Moreover, we have proposed the auxiliary density level estimation module (ADLE) to further help RAANet extract occlusion information of objects during training. Since RAACConv sets the channel number for each branch to the half of final output channels, and ADLE is not attached during inference, the proposed RAANet is able to run in real-time. Evaluations on nuScenes and KITTI datasets have shown that our proposed RAANet outperforms the SOTA baselines. We are also actively researching the generalization of the proposed range-aware module to transformer-based LiDAR detectors. We believe that the ongoing work in this area will further substantiate the robustness and versatility of our method.

References

[1] P. Bhattacharyya, C. Huang, and K. Czarnecki, “SA-Det3D: Self-attention based context-aware 3D object detection,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop, 2021, pp. 3022–3031.
[2] H. Caeser et al., “nuScenes: A multimodal dataset for autonomous driving,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2020, pp. 11618–11628.
[3] Q. Chen, L. Sun, E. Cheung, and Alan L. Yuille, “Every view counts: Cross-view consistency in 3D object detection with hybrid-cylindrical-spherical voxelization,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 21224–21235.
[4] 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., Springer, 2020, pp. 68–84.
[5] Q. Chen, S. Vora, and O. Beijbom, “PolarStream: Streaming LiDAR object detection and segmentation with polar pillars,” in Proc. Adv. Neural Inf. Process. Syst., 2021, pp. 26871–26883.
[6] 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., 2017, pp. 1907–1915.
[7] X. Chen, J. Wang, D. Held, and M. Hebert, “PanoNet3D: Combining semantic and geometric understanding for LiDAR point cloud detection,” in 2020 Int. Conf. 3D Vis., 2020, pp. 753–761.
[8] 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., 2023, pp. 13488–13498.
[9] Y. Chen, S. Liu, X. Shen, and J. Jia, “Fast point R-CNN,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2019, pp. 9775–9784.
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,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 2992–3001.

J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 779–788.

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, 2016.

S. He, Y. Wang, and H. Li, “POINTCNN: 3D object propositional generation and detection from point cloud,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2019, pp. 770–779.

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.

L. N. Smith and N. Topin, “Super-convergence: Very fast training of neural networks using large learning rates,” Artif. Intell. Mach. Learn. Multi-Domain Operations Appl., vol. 11006, 2019, Art. no. 1100612.

P. Sun et al., “RSN: Range sparse net for efficient, accurate LiDAR 3D object detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2021, pp. 5725–5734.

Yantao Lu (Member, IEEE) received the B.S. degree in electrical engineering from Xian Jiaotong University, Xi’an, China, in 2013, and the Ph.D. degree in electrical and computer engineering from Syracuse University, Syracuse, NY, USA, in 2020. He is currently an Associate Professor with the School of Computer Science, Northwestern Polytechnical University, Xi’an, China. His research interests include perception of autonomous driving and adversarial examples in deep neural networks.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Xuetao Hao received the B.S. degree in electrical engineering from Xidian University, Xi’an, China, in 2016, and the M.S. degree in electrical engineering from the University of Southern California, Los Angeles, CA, USA, in 2018. He is currently a Senior Engineer with Waymo LLC, Mountain View, CA, USA. His research interests include LLM, VLM, and perception of autonomous driving.

Yilan Li received the B.S. degree in information engineering from Xian Jiaotong University, Xi’an, China, in 2012, and the Ph.D. degree in electrical and computer engineering from Syracuse University, Syracuse, NY, USA, in 2021. She is currently an Instructor with the Xi’an University of Technology, Xi’an. Her research interests include neuromorphic systems for unmanned aerial vehicles planning and control and natural language processing.

Weiheng Chai received the bachelor’s degree in applied physics from the University of Science and Technology of China, Hefei, China, in 2016, and the Ph.D. degree in electrical and computer engineering from Syracuse University, Syracuse, NY, USA, in 2024. His research interest include autonomous driving and adversarial attacks.

Shiqi Sun received the B.S. degree in mechanical engineering from Beihang University, Beijing, China, in 2019, and the M.S. degree in mechanical engineering from Duke University, Durham, NC, USA, in 2021. He is currently working toward the Ph.D. degree with the School of Computer Science, Northwestern Polytechnical University, Xi’an, China. His research interests include perception in autonomous driving and robotic learning.

Senem Velipasalar (Senior Member, IEEE) received the B.S. degree (with Highest Hons.) in electrical and electronic engineering from Bogazici University, Istanbul, Turkey, in 1999, the M.S. degree in electrical sciences and computer engineering from Brown University, Providence, RI, USA, in 2001, and the M.A. and Ph.D. degrees in electrical engineering from Princeton University, Princeton, NJ, USA, in 2004 and 2007, respectively. Her research interests include mobile camera applications, wireless embedded smart cameras, multicamera tracking and surveillance systems, automatic event detection from videos, embedded computer vision, video/image processing, embedded smart camera systems, distributed multicamera systems, pattern recognition, and signal processing.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.