Image and Vision Computing 127 (2022) 104557

Contents lists available at ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

3D-VDNet: Exploiting the vertical distribution characteristics of point clouds for 3D object detection and augmentation

Weiping Xiao [a,b], Xiaomao Li [a,b], Chang Liu [a,b], Jiantao Gao [a,b], Jun Luo [a,b], Yan Peng [a,b], Yang Zhou [a,b,*]

[a] Shanghai University, Shanghai, China
[b] Engineering Research Center of Unmanned Intelligent Marine Equipment, Ministry of Education, Shanghai, China

ARTICLE INFO

Article history:
Received 25 June 2021
Received in revised form 25 July 2022
Accepted 9 September 2022
Available online 21 September 2022

Keywords:
3D object detection
Vertical distribution characteristics
Object augmentation
LiDAR point cloud

ABSTRACT

Accurate 3D object detection is limited by the sparsity of LiDAR-based point clouds. The vertical distribution characteristics (VDCs) of point clouds in pillars are robust to point-sparsity and provide informative semantic information on objects. Based on this, we propose a novel 3D object detection framework where the VDCs of point clouds are exploited to optimize feature extraction and object augmentation. More specifically, a Spatial Feature Aggregation module is proposed to perform robust feature extraction by decorating pillars with the VDCs. To spatially enhance semantic embeddings, we employ VDCs to construct a voxelized semantic map, acting as an additional input stream. Moreover, we develop an Adaptive Object Augmentation (AOA) paradigm, which adopts the VDC searching of suitable ground regions to "paste" virtual objects, thus avoiding conflicts with new scenes. Extensive experiments on the KITTI dataset demonstrate that our framework can significantly outperform the baseline, achieving 3.74%/1.59% moderate AP improvements on the Car 3D/BEV benchmarks with 38 FPS inference speed. Furthermore, we prove the stable performance of our AOA module across different detectors.

1. Introduction

LiDAR-based 3D object detection is a core component of environment perception systems for autonomous driving [1]. Given the sparsity and unevenness of point clouds, voxel-based methods strive to regularly organize point clouds through voxelization for efficient data processing. To effectively encode each voxel, the majority of voxel-based methods focus on the extraction of point cloud structural information [2–6], such as point-interactive features. However, the quality of these point-interaction features decreases with the gradual sparseness of the point clouds, which limits the detection performance accuracy.

A potential solution for the point-sparsity problem is to explore the overall distribution of point clouds rather than the information at the point-level, for example, employing the vertical distribution characteristics (VDCs) (i.e., maximum value (Max), minimum value (Min), mean value (Mean), and standard deviation (STD) of the point cloud z-axis coordinates) to decorate the pillars. We observe that the VDCs are insensitive to point-sparsity. It can also provide informative semantic information related to objects. As demonstrated in Fig. 1, the STD describes the difference between the ground and target region (where objects may exit), while other items in VDCs further indicate the specific differences in vertical space occupation of different objects (e.g., cars, buildings, and trees). These properties suggest the ability of VDCs to enhance feature extraction.

Moreover, the potential of VDCs in identifying the ground region allows us to further perform effective object augmentation in the training process. In real scenes, the objects should ideally be "pasted" on the ground. However, previous object augmentation methods [4] only copy-paste virtual objects from one scene into another, resulting in the pasted virtual object conflicting with the new scene. Fig. 2 reveals a conflict between pasted object #1 and the building, while pasted object #2 is located in the completely occluded region. This will undoubtedly hinder the recognition of objects.

Based on those investigations, we propose 3D-VDNet, a novel 3D object detection framework, to leverage the VDCs of point clouds for robust feature extraction and adaptive object augmentation. More specifically, a Spatial Feature Aggregation (SFA) module is designed to perform robust pillar feature extraction, whereby the pillars are decorated with VDCs to supplement the point-interactive features. Moreover, a voxelized semantic map is employed as an additional input to spatially enhance the semantic embedding of objects. This is performed by classifying each pillar into ground, target, and free (without point clouds) regions according to their VDCs. The pillar features and semantic maps are fed into the 2D Backbone and Detection head described in PointPillars [5] and the detection results are then output. Furthermore, we present an Adaptive Object Augmentation (AOA) paradigm that employs VDCs to adaptively search suitable ground regions to place virtual

* Corresponding author at: Shanghai University, Shanghai, China.
E-mail address: saber_mio@shu.edu.cn (Y. Zhou).

https://doi.org/10.1016/j.imavis.2022.104557
0262-8856/

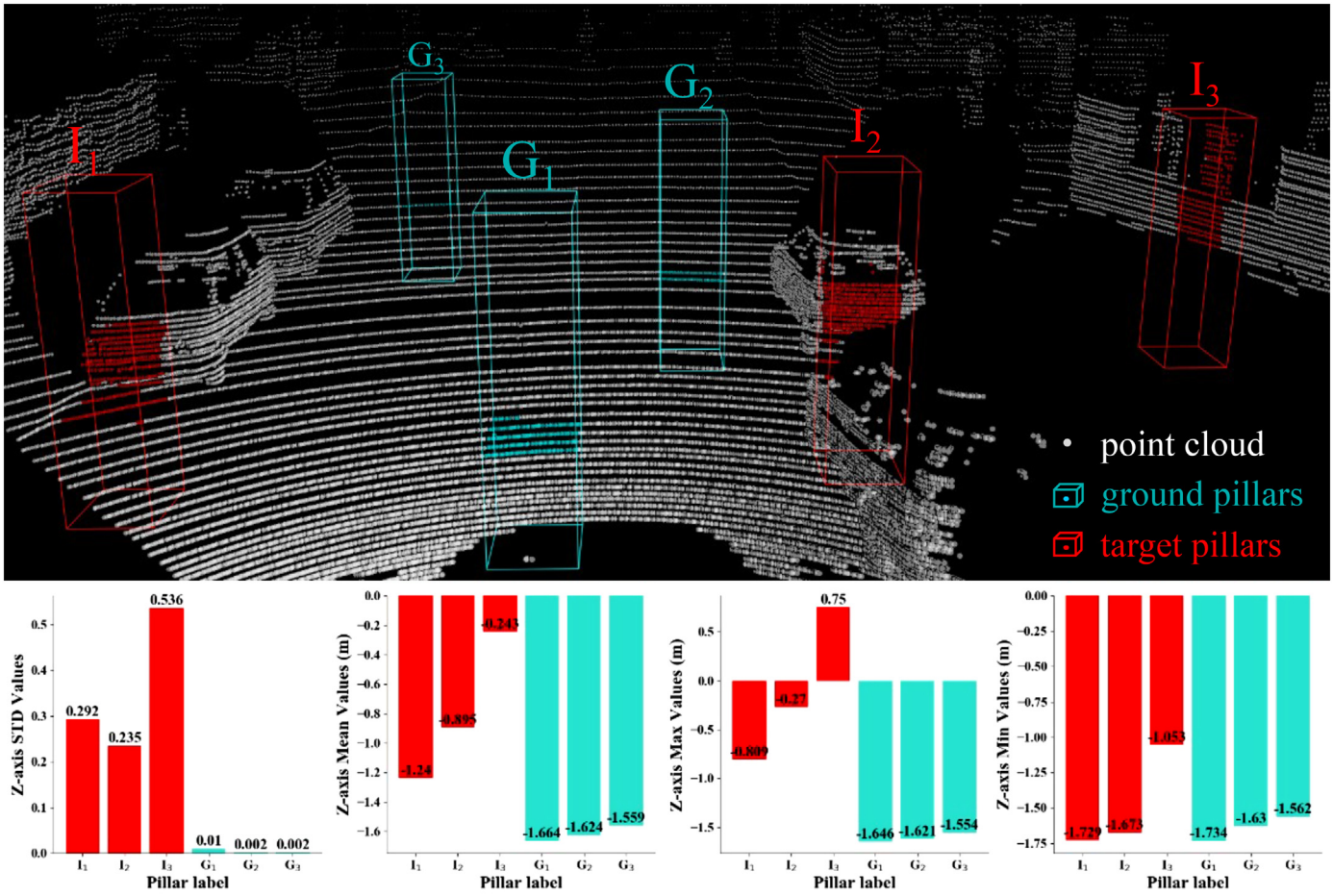

Fig. 1. Examples of point cloud VDCs within pillars. To aid visualization, a large voxel size of 1 × 1 × 4 m is used and inner point clouds are colored. *G* is the ground pillar; *I* is the target pillar; I_1 and I_2 are cars; and I_3 represents buildings.

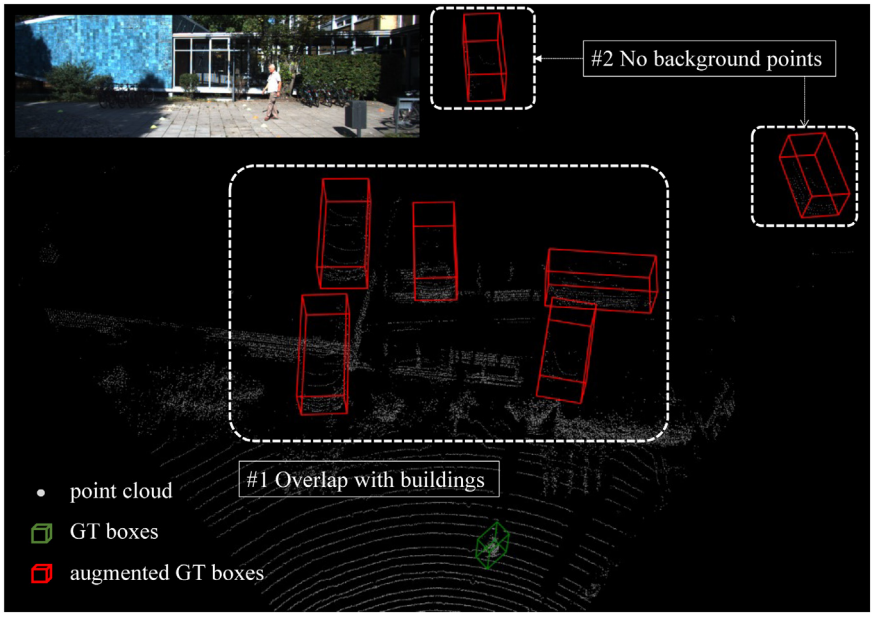

Fig. 2. Two problems of the copy-paste approaches employed by the previous object augmentation methods [4].

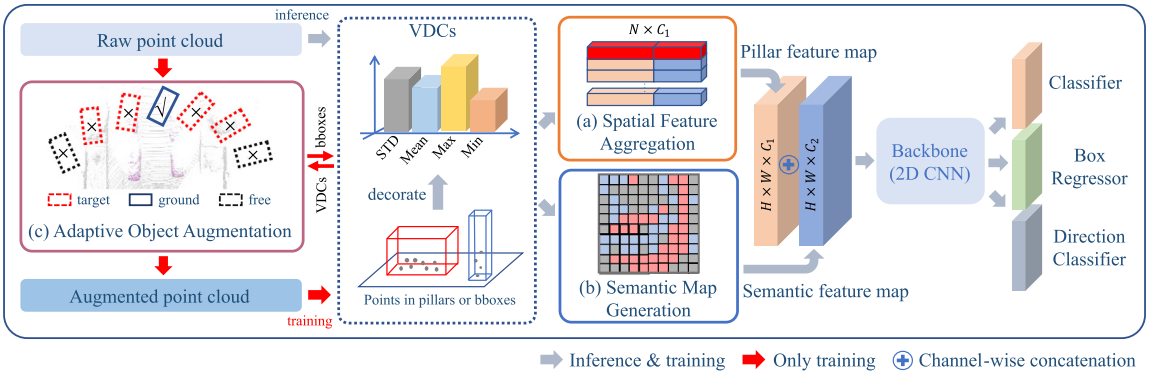

Fig. 3. Proposed 3D object detection framework. The three major components: (a) Spatial Feature Aggregation; (b) Semantic Map Generation; and (c) Adaptive Object Augmentation, all benefit from VDCs. Specifically, (a) encodes VDCs for robust and discriminative feature extraction. (b) employs VDCs to construct a voxelized semantic map for enhancing semantics embedding. Both the pillar and semantic feature maps are concatenated and pass through the Backbone & Detection head for prediction. During training, (c) adaptively augments virtual objects using the VDCs.

objects with the aim of avoiding conflicts with new scenes. Fig. 3 depicts the overall structure of the 3D-VDNet.

To evaluate the effectiveness of our method, we conduct extensive experiments on the challenging KITTI dataset. Results demonstrate our approach to significantly outperform the baseline, achieving 3.74% and 1.59% moderate (mod.) AP improvements on the Car 3D and bird's eye view (BEV) benchmarks, respectively. The key contributions of this work are summarized in the following.

(1) By leveraging the point cloud VDCs, we design a Spatial Feature Aggregation module that fuses the vertical distribution and point-interactive features to allow for robust feature extraction.

(2) The VDCs are further utilized to create a voxelized semantic map as a further input, which enhances the semantic embedding of the objects.
(3) An adaptive object augmentation paradigm is proposed to overcome the conflict between augmented objects and corresponding scenes via the point cloud VDCs.
(4) Extensive experiments demonstrate the effectiveness of the point cloud VDCs. In addition, the proposed AOA module can act as a plug-and-play component, with a stable performance across detector types (voxel- or point-based).

2. Related work

2.1. LiDAR-based 3D object detection

LiDAR-based 3D object detection can generally be categorized into point- and voxel-based methods. [7] provides a comprehensive introduction of LiDAR-based 3D object detection approaches, below we describe the two methods in detail.

Point-based Methods. For the powerful feature learning capability of PointNet/PointNet++ [8,9] in the classification and segmentation domain, several studies have provided extensions to point-based 3D object detection. F-PointNets [10] and IPOD [11] obtain 2D results to crop point clouds and subsequently following the implementation of PointNet to aggregate features for 3D bounding box (bbox) predictions. PointRCNN [12] is a pioneering framework that avoids relying on 2D results, rather it employs PointNet++ to directly generate 3D proposals from the raw point cloud. STD [13] attempts to refine the bounding boxes (bboxes) in a sparse-to-dense manner. VoteNet [14] implements the deep Hough voting strategy to improve the feature aggregation of the object. Due to the irregularity of point clouds and the large amounts of data, point-based methods are computationally expensive, while our method follows the voxel-based setting.

Voxel-based Methods. To effectively process the sparse and irregular data of the point cloud, the majority of existing studies attempt to regularly organize the point cloud via voxelization and the subsequent implementation of advanced 2D/3D convolutional neural networks (CNN) for predictions [15]. VoxelNet [4] is the first end-to-end learning framework that unifies feature extraction and bbox prediction using 3D CNN. However, the 3D convolutional layer proves to be computationally expensive. Based on the sparsity of non-empty voxels, SECOND [4] introduces 3D sparse convolution to improve the processing efficiency of the 3D voxel. Furthermore, PointPillars [5] removes the 3D CNN operation by simplifying SECOND, dynamically converting point clouds into pillars to construct a 2D pseudo image. Inspired by the conversing of point clouds into pillars by PointPillars, our work further explores the VDCs of point clouds in order to perform robust feature extraction and adaptive object augmentation. Several recent studies [16,17] merge voxel- and point-wise features to generate more informative 3D features compared to previous methods. However, they are two-stage networks that require more computing resources compared to our proposed one-stage detector.

2.2. Representation learning on voxels

Several traditional methods employ handcrafted features to represent voxels. For example, [18,19] uses six statistical quantities to encode non-empty voxels, while [20–22] encodes each voxel as occupancy, truncated signed, or binary. These handcrafted features are typically designed for specific scenarios and tasks, and thus may not be generalizable to variable environments such as autonomous driving. VoxelNet utilizes a tinny PointNet to generate learned features for each voxel, whereby the point cloud is initially decorated with coordinate (x, y, z), intensity r, and distance from the cluster center $(\triangle x_c, \triangle y_c, \triangle y_c)$. Due to the great flexibility of PointNet to generate point-wise features for voxel representation, the majority of voxel-based studies adopt similar strategies [3–5], yet the point decoration is distinct. For example, PointPillars includes the distance from the pillar center $(\triangle x_p, \triangle y_p)$ to standardize the local background of the point. These methods focus on extracting point-interactive features to represent voxels, and the quality of these features varies with the point cloud sparseness. Our proposed SFA module extracts robust features by introducing the VDCs.

Recently, WYSIWYG [2] proposed a new form of voxel representation that employs the raycasting mechanism, labeling each voxel with visibility to distinguish between the lidar-perceived and obscured regions. In contrast, our method employs VDCs to label each pillar as a ground, target, or free region. This is computationally friendly and provides semantic clues to facilitate object detection.

2.3. 3D object augmentation

3D Object augmentation, a novel form of data augmentation that enriches data training, was initially proposed in SECOND [4]. This technology is currently widely employed in state-of-the-art detectors [2,4,5,12,13,17], significantly improving convergence speed and performance. However, its copy-pastes technique may result in the overlapping of the augmented objects with those in the new scene (e.g., buildings and trees), thus hindering object recognition. Recently, WYSIWYG [2] used the raycasting principle to remove the point cloud that blocks the augmented 3D object to make it satisfy the visibility reasoning. Since the nature of this method is still a copy-pastes manner, and hence the overlap between the augmented object and new scene in the vertical direction is still unavoidable. In the current paper, we propose a simple and yet reasonable VDC-based object augmentation method that adaptively searches suitable ground regions to augment the target objects.

3. 3D-VDNet detector

In this section, we first introduce the preliminaries of pillar-based 3D object detection and subsequently describe the proposed framework.

3.1. Preliminaries

Based on the PointPillars settings, the input LiDAR point clouds with initial features $f_i = \{x, y, z, r\}$ are voxelized into pillars of size $D \times P \times N$, where D denotes the point dimension, P indicates the number of non-empty pillars, and N is the number of points in each pillar. For each non-empty pillar, if the number of inner points is greater than N, then random sampling is performed, while if it is less than N, zero-padding is applied.

Each point p_i within the pillar v_j is augmented with interactive features as follows:

$$f_i^{point} = \left\{\Delta x_c, \Delta y_c, \Delta z_c, \Delta x_p, \Delta y_p\right\}, \tag{1}$$

where subscript c denotes distance to the arithmetic mean of all points in the pillar and subscript p indicates the offset from the pillar center. These point-wise features are aggregated in a pillar by employing a tinny PointNet (TPN) to obtain learned pillar-wise point-interactive features $f^{interactive}$ of size $C_i \times P$, where

$$f^{interactive} = TPN\left(\left\{\left(f_i \cup f_i^{point}\right) | \forall p_i \in v_j\right\}_{j=0}^{P-1}\right). \tag{2}$$

The point-interactive features are then scattered back to the original pillar locations through mapping function F_s to create a pillar feature map $F^{interactive}$ of size $C_i \times H \times W$, where H and W indicate the height and width of the BEV point cloud range.

Finally, the obtained pillar feature map $F^{interactive}$ passes through a 2D CNN backbone network and detection head for classification confidence predictions and 3D bbox regression.

3.2. Overall framework

Our framework builds on pillar-based 3D object detection (Fig. 3), whereby the input point clouds are voxelized into pillars. The Spatial Feature Aggregation module is then used to encode each non-empty pillar as points and VDCs to produce a pillar feature map. As an additional input stream, the VDCs are employed to initially label each pillar as a ground, target, or free region, constructing a semantic feature map. Two feature map types are subsequently concatenated and pass through the Backbone & Detection head for confidence prediction and bbox regression. During training, the raw point cloud is adaptively augmented with the virtual objects via our proposed object augmentation paradigm, where the VDCs serve as an indicator.

3.3. Spatial feature aggregation

The accurate regression and classification of 3D objects from point clouds rely on a robust and discriminative feature extraction. The point-interaction features preserve the fine information, while the quality is reduced with the gradual sparseness of point clouds. In contrast, the pillar VDC is robust to the sparsity of the point cloud, yet it is considered as "rough". This motivates us to fuse both approaches (Fig. 4).

We denote $F_{v\in\{STD,Mean,Min,Max\}}(\{p_i^{(z)}\}_{i=1}^{n})$ as a statistical function employed to calculate the point cloud VDCs in a pillar, where $p_i^{(z)}$ is the z-axis coordinate of a point and n is the number of points (we set $F_{STD} = 0$, when $n = 1$). As described in Section 3.1, pillars with sparse points will undergo zero-padding during the voxelization process, which consequently affects the true value of the VDCs. Therefore, we employ a recorded number of non-zero points n_j to restore the real points in a pillar. A fully connected network (FCN) consisting of a linear layer, a batch normalization (BatchNorm) layer, and a rectified linear unit (ReLU) layer is used to extract the vertical distribution feature $f^{distribution}$ of size $C_d \times P$ from the VDCs, where

$$f^{distribution} = FCN\left(\left\{F_{v\in\{STD,Min,Max,Mean\}}\left(\left\{p_i^{(z)}\right\}_{i=0}^{n_j-1}\right)\right\}_{j=1}^{P}\right). \tag{3}$$

The vertical distribution and point-interaction features are concatenated to produce a pillar feature map of size $(c_i + c_d) \times H \times W$ (Fig. 4), where $(c_i + c_d)$ are the total feature channels. Hyperparameter $p = c_i/c_d$ controls the proportion of the two types of feature channels (see Section 4.4.1 for more details on the ablation study).

Through the proposed SFA module, we are able to extract more robust and discriminative features with fine and rough spatial characteristics for the accurate prediction of bboxes and precise classification.

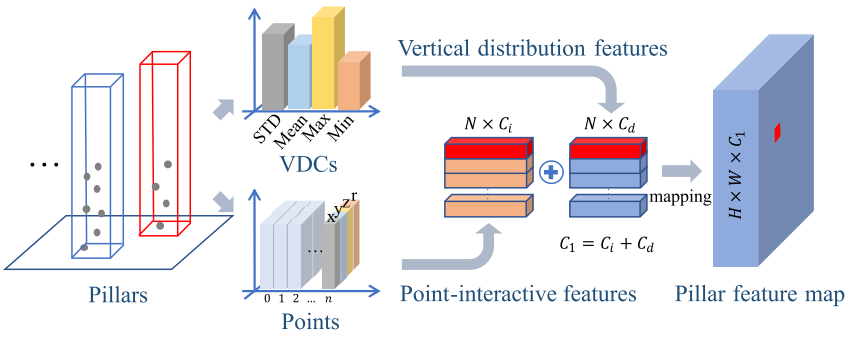

Fig. 4. Illustration of the spatial feature aggregation module.

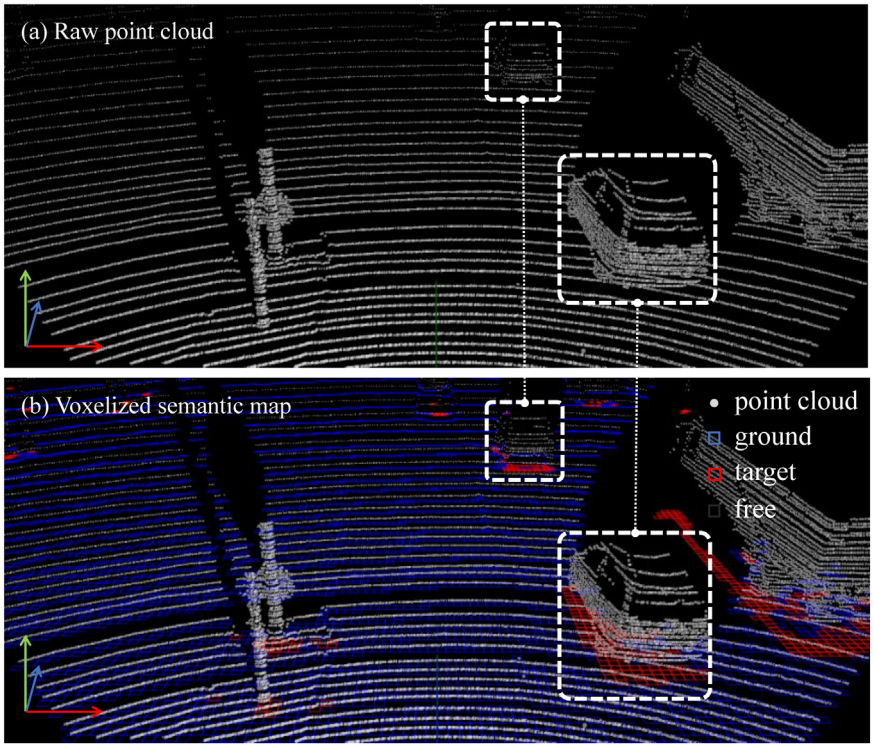

Fig. 5. Relationship between (a) raw point clouds and (b) voxelized semantic map. Blue, red, and black grids denote ground, target, and free regions, respectively.

3.4. Semantic map generation

The pillar VDC can help to determine the potential location of objects from a large point clouds scene. Therefore, we construct a voxelized semantic map by labeling each pillar as a ground, target, or free region (Fig. 5), which consequently generate semantic map feature improving the object discrimination feature capability.

Specifically, let $v_{(i,j)} = \{p_i\}$ denote a pillar with index (i,j) in the pseudo-image. For non-empty pillars, we can use a threshold t_{std} to label them as ground or target regions due to the significant difference in STD between the point clouds of these regions (Fig. 1). Empty pillars are labeled as free regions. Therefore, the initial label L of pillars can be described as:

$$L = \begin{cases} ground, & if\ 0 \leqslant F^{(z)}_{STD}(v_{i,j}) \leqslant t_{std} \\ target, & if\ F^{(z)}_{STD}(v_{i,j}) > t_{std} \\ free, & if\ v_{i,j}\ is\ empty \end{cases}, \tag{4}$$

where the threshold t_{std} is set as 0.01. However, a few non-empty pillars near the object center are incorrectly classified as ground pillars (Fig. 6 (a)). Although the point clouds close to the object center and the ground point clouds are observed to have a similar distribution, i.e., little varying on the z-axis, their vertical spatial occupations are distinct. To this end, we designed a label rectification sub-module to rectify the pillar labels. More specifically, we customize a $k \times k$ filter to collect those ground pillars L_g that are close to the target pillars (Fig. 6(b)). Height threshold t_{max} is then employed to select potential target pillars (Fig. 6 (c)) from the collected ground pillars:

$$L_g = \begin{cases} target, & if\ F^{(z)}_{Max}(v_{i,j}) \geqslant t_{\max} \\ ground, & otherwise \end{cases}. \tag{5}$$

In practice, we implement this process efficiently through a convolutional operation, whereby threshold t_{max} is selected as the average height of the GT bbox centers in *val* set.

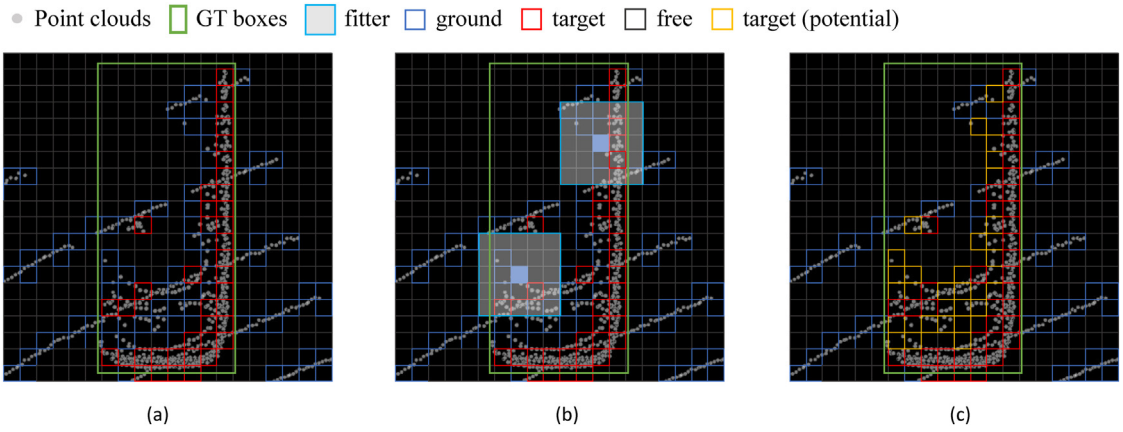

Fig. 6. Illustration of the label rectification module. (a) Initial pillar labels. (b) Collection of ground pillars L_g close to the target pillars by the customized filter. (c) Rectified pillar labels. Through such a process, a few incorrectly classified ground pillars in GT bboxes (blue grids) are rectified as potential target pillars (yellow grids).

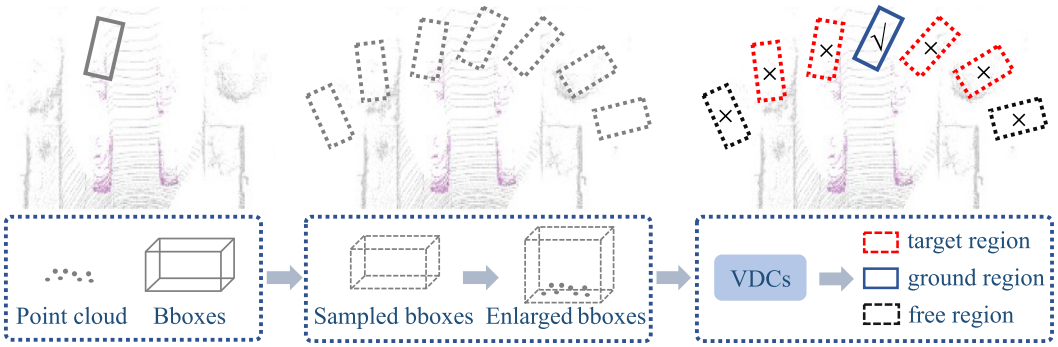

Fig. 7. The schematic diagram of the adaptive object augmentation module.

The 2D spatial size of the voxelized semantic map coincided with that of the pillar feature map, allowing for its incorporation into our detector via several potential options. For example, we can directly concatenate and feed the semantic map into a backbone network; this is simple. Another strategy is to employ a one-layer FCN or CNN to obtain the semantic feature map prior to concatenating the pillar feature map. Our lightweight Semantic Map Generation (SMG) module is able to spatially enhance the semantic embedding concisely. The SMG module is discussed in more detail in Section 4.4.2.

3.5. Adaptive object augmentation

Prior studies augment virtual objects in a copy-paste manner, ignoring the new scene structure. As illustrated in Fig. 2, the pasted virtual object may overlap with a wall or may be placed in a completely occluded region, thus hindering object recognition. For autonomous driving scenarios, we expect that the augmented object is only "pasted" on the ground region, which has the following advantages: 1) the augmented object does not conflict with the new scene; and 2) the surroundings of the augmented object contain appropriate context information to benefit object detection. However, determining a suitable ground region to "paste" the virtual object without the point cloud semantic information proves to be a difficult task.

As the VDCs can identify the ground region, we can use them to determine whether the region is suitable for augmenting virtual objects. To obtain the VDCs of a sampled location, we enlarge the sampled ground truth (GT) bbox (without the sampled points) into a large "pillar" along the z-axis that is then used to collect the point clouds for the VDC calculation. Following this, we must determine how to "paste" the sampled GT bboxes. As the global rotation of a bbox will not change its relative position to the sensor, we can apply this to generate bbox copies, which can evenly cover the effective point cloud range. The schematic diagram of the AOA module depicts in Fig. 7.

Algorithm 1 Adaptive Object Augmentation.

Input:
P is a point cloud with size $(k, 4)$
S is a set of sampled GT bboxes with size $(n_{in}, 7)$
ϕ is an object augmentation range with a default value of $[-\pi/5.5, \pi/5.5]$
n is the number of copy-paste times with a default value of 10
t_g is the ground region threshold with a default value of 0.08
t_n is the number of point clouds threshold with a default value of 5
Output:
R is a set of resampled GT bboxes with size $(n_{out}, 7)$

Rotate angular intervals $d_\phi \leftarrow (\phi[1] - \phi[0])/n$;
Rotate boxes S *to left boundary* $\phi[0]$ *around sensor origin:* $I \leftarrow rotate(S, \phi[0])$;
foreach i *in* n **do** paste boxes $P[i] \leftarrow rotate(I, \phi[0] + d_\phi * i)$;
Compute IoU between P *and* G: $D_o \leftarrow IoU_{BEV}(P, G)$;
Select the index of boxes from P *that do not overlap with* G: $idx_1 \leftarrow D_o = 0$;
Compute STD of enlarged P: $V_{STD} = F^{(z)}_{STD}(P_{enlarged})$;
Select the index of boxes from P *that are pasted in the ground region:* $idx_2 \leftarrow V_{STD} < t_g$;
Obtain candidate boxes: $C \leftarrow P[idx_1 \bigcap idx_2]$;
Update boxes C *that contain appropriate context information:* $C \leftarrow C[Cnt(C_{enlarged}) > t_n]$;
for $j \leftarrow 0$ *to* n_{in} **do**
  Select the copies belonging to S_j *from candidate boxes* C: $B_j \leftarrow C[copies\ of\ S_j]$;
  if B_j *is not None* **then**
    if R *is None* **then**
      Select one box randomly from boxes B_j: $R[j] \leftarrow random(B_j)$;
    else
      Compute IoU between B_j *and* R: $D_{cj} \leftarrow IoU_{BEV}(B_j, R)$;
      Obtain the candidate boxes from B_j *that do not overlap with* R: $D_j \leftarrow B_j[D_{cj} = 0]$;
      Select one box randomly from candidate boxes D_j: $R[j] \leftarrow random(D_j)$;
    end
  else
    $R[j] \leftarrow None$;
  end
end
Obtain the resampled GT bboxes from R: $R[:] \leftarrow R[R \neq None]$;
return R;

Algorithm 1 details the adaptive object augmentation paradigm. In Lines 1 to 3, we rotate the sampled GT bboxes G around the sensor to the left boundary of the point clouds and subsequently rotate and paste them clockwise at fixed angular intervals until the whole point cloud range is covered. Following this, we select candidate boxes C from pasted boxes P according to the following criteria (Lines 4 to 9): 1) do not overlap the GT bboxes; 2) on the ground region; and 3) contains appropriate context information (at least t_n points in the ground region). Here, using the STD value in the VDCs as an indicator can effectively distinguish the ground region as the enlarged bbox contains a large point cloud range compared to the unit pillar. For each sampled GT bbox $S[j]$, we find the corresponding copies B_j from the candidate bboxes C and select one as the final resampled GT bbox $R[i]$ (Lines 10 to 25).

This simple yet effective process allows for the sampled GT bboxes to be pasted to a more suitable ground region, achieving adaptive object augmentation in the new scene. Furthermore, this object augmentation paradigm does not require any preprocessing, ensuring suitability for both point- and voxel-based detectors. Section 4.4.3 provides more details of the experiments.

3.6. Loss function

We employ the same loss function as SECOND [4] and PointPillars [5]. In particular, the object classification adopts focal loss $\mathcal{L}_{cls}$, with $\alpha = 0.25$ and $\gamma = 2$ as recommended by [23], while the localization regression adopts smooth L1 loss $\mathcal{L}_{loc}$ (i.e., Huber loss), with $\sigma = 3.0$. Furthermore, the cross-entropy loss $\mathcal{L}_{dir}$ is used for the direction classification. The objective of the three tasks is therefore

$$\mathcal{L}_{lotal} = \frac{1}{N_{pos}}(\beta_1 \mathcal{L}_{cls} + \beta_2 \mathcal{L}_{loc} + \beta_3 \mathcal{L}_{dir}) \tag{6}$$

where N_{pos} is the number of positive anchors and β are the constant factors of loss terms. We set $\beta_1 = 2.0, \beta_2 = 1.0$ and $\beta_3 = 0.2$.

4. Experiments

We evaluate the proposed 3D-VDNet on the challenging KITTI 3D/BEV detection benchmark [24]. In this section, we first introduce the dataset, evaluation metrics, and the implementation of our method and subsequently compare the proposed 3D-VDNet with state-of-the-art 3D detection methods. Furthermore, we perform extensive ablation studies to investigate the individual components of our methods.

4.1. Datasets and evaluation

The KITTI 3D object detection benchmark contains 7,481 annotated LiDAR point clouds with 3D bboxes for training and 7,518 LiDAR point clouds for testing. The training samples are divided into *train* split (3,712 samples) and *val* split (3,769 samples) following the common protocol described in [25,26]. The ablation studies are performed on this *train*/*val* split with the most commonly used Car class. For a fair comparison on the test set, we train the model on re-split *train*/*val* sets according to [5]. We use average precision (AP) based on 40 recall positions with an (IoU) threshold of 0.7 as an evaluation metric following the official KITTI evaluation protocol. We adopt the mean AP (mAP) to evaluate the overall performance of the three difficulty levels (easy, moderate, and hard).

4.2. Implementation details

Pre-processing. To align the network input, we set the point cloud range as [0, 69.12], [−39.68, 39.68], [−3, 1] and the voxel size as (0.16, 0.16, 4) m along the x, y, and z-axes, respectively. We remove the points that are invisible in the image view. Through voxelization, each non-empty pillar resample contains 32 points.

Network details. Our SFA module operates over each non-empty pillar to produce a 64-d feature vector in which the proportion of the point-interactive and vertical distribution feature channels is 40/24. This feature vector is then scattered back to the original pillar locations, generating a $64 \times 432 \times 496$ pillar feature map. The SMG module takes the VDCs of pillars as the input and transforms them into pillar labels. A single-layer CNN (3×3 convolution layer followed by BatchNorm and ReLU) is then adopted by SMG to generate a $32 \times 432 \times 496$ semantic feature map. Two feature map types are subsequently concatenated into a $96 \times 432 \times 496$ feature map and pass through the Backbone network. The backbone network consists of three blocks of fully convolutional layers as for PointPillars. Each block has convolutional layers and deconvolutional layers. Fig. 8 details the backbone network. Other network settings follow PointPillars.

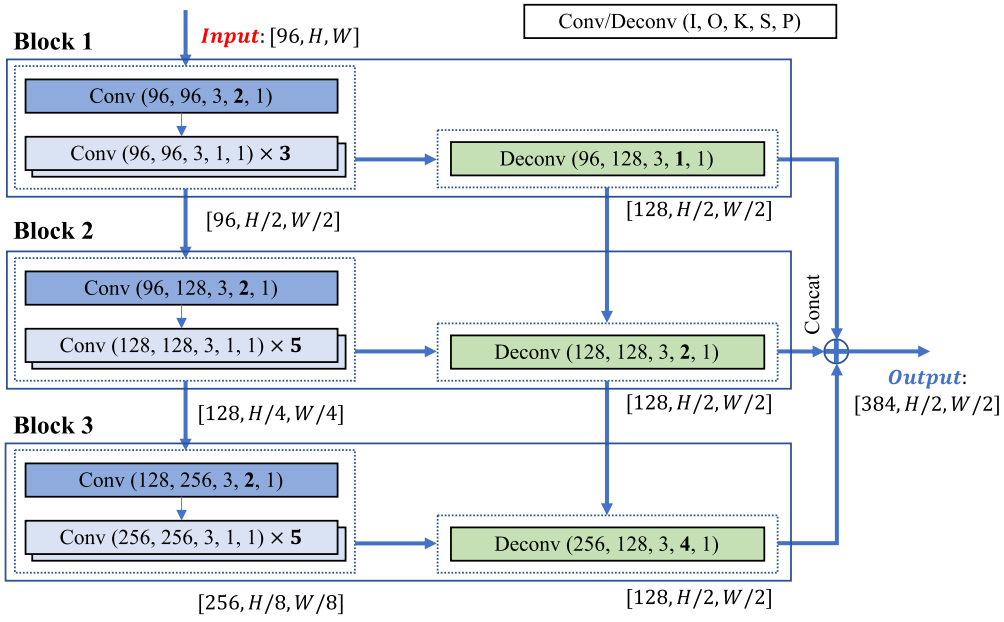

Fig. 8. Details of the backbone network. I, O, K, S, and P denote input channel, output channel, kernel size, stride, and padding.

Table 1
3D object detection performance on the KITTI test set. "L", "I" and "I + L" indicate the application of LiDAR point clouds, RGB images, and a fusion of the two, respectively. Numbers in italics denote the optimal results for one-stage detectors, while numbers in bold highlight the best-performing detectors.

Method	Input	Stage	FPS(HZ)	3D Detection			BEV Detection		
				Easy	Mod.	Hard	Easy	Mod.	Hard
MV3D [25]	I+L	Two	2.8	74.97	63.63	54.0	86.62	78.93	69.8
AVOD [32]	I+L	Two	10.0	83.07	71.76	65.73	90.99	84.82	79.62
F-PointNet [10]	I+L	Two	5.9	82.19	69.79	60.59	91.17	84.67	74.77
F-ConvNet [33]	I+L	Two	2.1	87.36	76.39	66.69	91.51	85.84	76.11
MMF [31]	I+L	Two	12.5	**88.4**	77.43	70.22	93.67	**88.21**	81.99
PointRCNN [12]	L	Two	8.9	86.96	75.64	70.7	92.13	87.39	82.72
Fast Point R-CNN [17]	L	Two	15.0	85.29	77.4	70.24	90.87	87.84	80.52
Part A^2 Net [30]	L	Two	14	87.81	**78.49**	**73.51**	91.7	87.79	84.61
ComplexYOLO [34]	L	One	15.6	55.93	47.34	42.6	77.24	68.96	64.95
VoxelNet [3]	L	One	4.4	77.82	64.17	57.51	87.95	78.39	71.29
SECOND [4]	L	One	20.0	83.34	72.55	65.82	89.39	83.77	78.59
PointPillars [5]	L	One	42.4	82.58	74.31	68.99	90.07	86.56	82.81
SARPNET [6]	L	One	-	85.63	76.64	71.31	92.21	86.92	81.68
ContFuse [35]	I+L	One	16.7	83.68	68.78	61.67	**94.07**	85.35	75.88
3D-VDNet	L	One	38.0	***87.13***	***78.05***	***72.9***	91.72	***88.15***	**84.65**

Table 2
Performance of the proposed method with different configurations. The 3D/BEV detection AP on Car class for easy, moderate, and hard subsets on KITTI *val* split is reported. AOA, SMG, and SFA refer to the Adaptive Object Augmentation, Semantic Map Generation, and Spatial Feature Aggregation modules, respectively.

	AOA	SMG	SFA	ATSS	3D Detection (Car)			BEV Detection (Car)		
					Easy	Mod.	Hard	Easy	Mod.	Hard
(a)					87.94	78.63	75.81	91.7	87.89	86.81
(b)	✓				88.44/+0.5	79.46/+0.83	78.16/+2.35	94.04/+2.34	88.87/+0.98	88.16/+1.35
(c)		✓			88.41/+0.47	80.4/+1.77	77.76/+1.95	92.19/+0.49	88.82/+0.39	87.45/+0.64
(d)			✓		87.69/−0.25	80.33/+1.7	77.64/+1.83	92.01/+0.31	89.71/+1.82	87.56/+0.75
(e)	✓	✓			89.36/+1.42	80.72/+2.09	78.11/+2.3	93.83/+2.13	90.12/+2.23	87.83/+1.02
(f)	✓	✓	✓		89.34/+1.4	81.14/+2.52	78.57/+2.76	93.39/+1.69	90.09/+2.2	87.97/+1.16
(g)	✓	✓	✓	✓	**90.15/+2.21**	**81.66/+3.03**	**78.97/+3.16**	**94.12/+2.42**	**90.71/+2.82**	**88.35/+1.54**

Training and inference details. We adopt the Adam optimizer [27] and one-cycle scheduler [28] with a maximum learning rate of 3e-3, division factor of 10, momentum range of [0.95, 0.85] and weight decay of 0.01. We employ the regression target assignment strategy of ATSS [29]. We train all our models for 85 epochs with a batch size of 16 that is equally distributed on 4 GPU cards (1080Ti). For inference, we filter out the low-confidence bboxes with a threshold equal to 0.3. Finally, we apply the rotated NMS with a threshold of 0.01 to remove redundant boxes and generate the final 3D detection results.

Data augmentation. In addition to the proposed adaptive object augmentation strategy, we adopt an additional three data augmentation strategies on the KITTI dataset to prevent overfitting: 1) randomly flipping along the x-axis; 2) randomly rotating around the z-axis with the range $[-\pi/4, \pi/4]$; and 3) rescaling with a scale factor sampled from [0.95, 1.05].

4.3. Main results

To facilitate comparisons with other state-of-the-art approaches, we submit the results of our 3D-VDNet to the KITTI test server[1] (Table 1). Our approach is observed to outperform the baseline by a large margin. More specifically, our model leads the PointPillars [5] by (4.55%, 3.74%, 3.91%) AP in the 3D detection and (1.65%, 1.59%, 1.84%) AP in the BEV detection. Under the "moderate" difficultly level, our method surpasses all the one-stage and the majority of the two-stage approaches, including SHAPNET [6] (by 1.41% AP) and PointRCNN [12] (by 2.41% AP) in the 3D detection. In the BEV detection, our method exceeds all LiDAR-only approaches, for example the two-stage approach Part A^2 Net [30] by 0.36% AP. 3D-VDNet is still able to maintain a comparable performance to the state-of-the-art multi-sensors two-stage method MMF [31] in the 3D/BEV detection (78.05% vs 77.43%/88.15% vs 88.21%). For the "hard" level cases, our method suppresses all other approaches in the 3D/BEV detection, with the exception of Part A^2 Net in the 3D detection. This indicates the effectiveness of our proposed feature extraction strategy and the reasonable adaptive object augmentation paradigm. In addition, our 3D-VDNet can run at 38 FPS which is faster than most of the methods. Fig. 11 presents exemplary qualitative results, demonstrating the effectiveness of our method.

4.4. Ablation studies

We conduct a comprehensive analysis of the effectiveness of our proposed modules. As reported in Table 2, each proposed module significantly outperforms its counterpart (b, c, d), and their combinations (e, f) booms the baseline (a) by a large margin. Moreover, the introduced ATSS target assignment strategy further improves the final performance (g). In the following, we describe the ablation details of each proposed module.

4.4.1. Effect of spatial feature aggregation module

We present two types of VDC encoding strategies while maintaining all other settings the same as in the previous experiments. In Table 3, VFE refers to the voxel feature encoding layer described in [4,5], which employs a TPN layer product point-interactive features. VFE + VDCs denotes the fusion of the VDCs to each initial feature $f_i = \{x, y, z, r\}$ within a pillar, and then a TPN layer is employed to generate enhanced point-interactive features. To achieve this, the 1D vector VDCs needs to expand from $(1, 4)$ to $(n, 4)$, where n is the number of points within a pillar.

[1] http://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=3d.

Table 3
Effect of several feature encoding strategies.

Method	3D Detection (Car)			BEV Detection (Car)		
	Easy	Mod.	Hard	Easy	Mod.	Hard
Ours w VFE	88.6	79.82	78.66	92.76	89	88.3
Ours w VFE + VDCs	88.39	79.71	78.67	93.91	88.89	88.26
Ours w SFA	**90.15**	**81.66**	**78.97**	**94.12**	**90.71**	**88.35**

In SFA, an additional FCN layer is employed to encode the VDCs to generate the vertical distribution features. This is then concatenated with the point-interactive features. The proportion of the two feature channels is set as $p = 40/24$. The results in Table 3 demonstrate that using an additional FCN layer to encode the VDCs (SFA) can better explore the potential of the VDCs.

Furthermore, we reveal the importance of the vertical distribution features obtained from the VDCs by adjusting proportion p. The optimal combination for the 3D and BEV detections is observed as $p = 40/24$. No significant reductions are observed in the performance when the proportion is gradually reduced to $p = 8/56$ (Fig. 9). A significant decline is observed when completely replacing the point-interactive features with the vertical distribution features ($p = -/64$). This may be attributed to the lack of location information (i.e., x and y) contained in the VDCs. Therefore, we employ both the pillar center (x, y) and the VDCs as the FCN input to obtain the enhanced vertical distribution features, denoted as ($p = -/64'$). The fusion of our approach with SFA ($p = -/64'$) outperforms that with SFA ($p = 64/-$), i.e., VFE, further confirming that the vertical distribution features provide sufficient critical information for object detection.

4.4.2. Effect of semantic map generation module

We present three types of semantic map generation strategies. Table 4 reports the performance of our approach fused with these strategies. The learned semantic feature map (through CNN or FCN) is observed to be superior to directly use the raw voxelized semantic map. Moreover, integrating our approach with SMG (CNN) is slightly better than with SMG (FCN). This is attributed to the enhancement of the feature representation ability due to the convolution operation with a proper receptive field. Table 5 investigates the effectiveness of the label rectification (LR) sub-module in the SMG. The fusion of our approaches with the LR sub-module outperforms the corresponding approaches without the LR sub-module, and with $k = 5$ achieving the

Table 4
Effect of several semantic map generation strategies. SMG (raw): raw voxelized semantic map; SMG (FCN): semantic feature map generated by an FCN layer; and SMG (CNN): semantic feature map generated by a CNN layer.

Method	3D Detection (Car)			BEV Detection (Car)		
	Easy	Mod.	Hard	Easy	Mod.	Hard
Ours w SMG (raw)	88.87	79.53	76.78	92.85	88.69	87.94
Ours w SMG (FCN)	88.74	81.58	**78.98**	92.6	90.6	**88.4**
Ours w SMG (CNN)	**90.15**	**81.66**	78.97	**94.12**	**90.71**	88.35

Table 5
Effect of label rectification sub-module in SMG. Hyperparameter k represents the size of the customized convolution kernel.

Method	3D Detection (Car)			BEV Detection (Car)		
	Easy	Mod.	Hard	Easy	Mod.	Hard
Ours w/o LR	88.33	80.77	78.34	92.32	88.56	87.84
Ours w LR ($k = 3$)	88.66	81.35	78.85	92.51	88.77	88.0
Ours w LR ($k = 5$)	**90.15**	**81.66**	**78.97**	**94.12**	**90.71**	**88.35**
Ours w LR ($k = 7$)	90.01	81.0	78.62	93.66	89.78	88.07

Table 6
Effect of different voxel size v on semantic information representation. $v = 0.16^2$ is the default setting of our experiments.

Method	3D Detection (Car)			BEV Detection (Car)		
	Easy	Mod.	Hard	Easy	Mod.	Hard
Ours ($v = 0.1^2$)	**90.96**	81.89	79.41	93.15	90.39	88.37
Ours ($v = 0.12^2$)	90.58	**82.27**	**79.69**	93.05	89.25	**88.66**
Ours ($v = 0.14^2$)	89.99	81.64	79.05	92.87	90.33	88.33
Ours ($v = 0.16^2$)	90.15	81.66	78.97	**94.12**	**90.71**	88.35
Ours ($v = 0.18^2$)	87.95	80.39	78.02	93.82	89.94	87.97
Ours ($v = 0.2^2$)	88.66	78.91	77.35	93.04	89.74	87.65

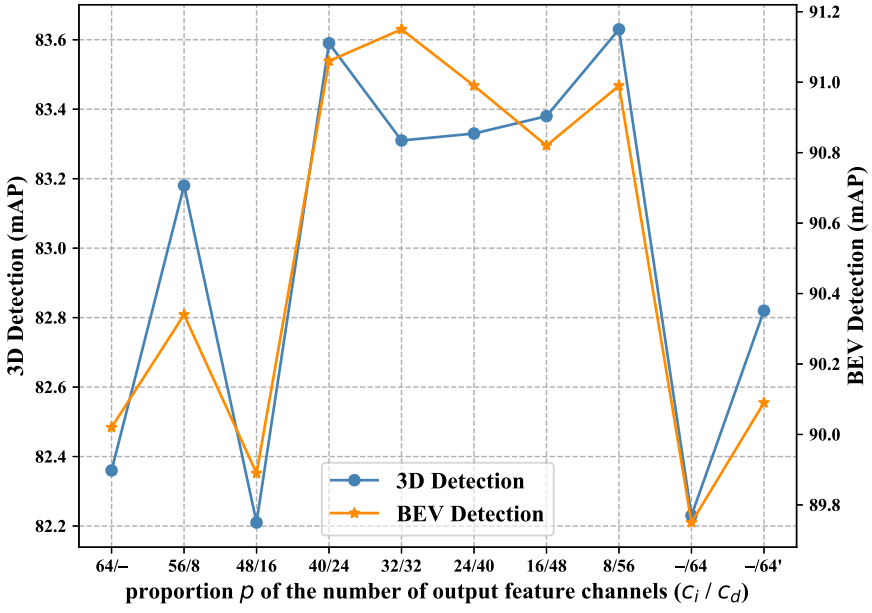

Fig. 9. Performances of SFA module for varying proportion p on KITTI *val*. $p = 40/24$ denotes the point-interactive and vertical distribution feature channels of 40 and 24, respectively. $p = 64/-$, i.e., VFE, denotes only using the point-interactive features while $p = -/64$ denotes only using the vertical distribution features. Special case $p = -/64'$ denotes the enhanced vertical distribution features that include location information.

Table 7
Effect of different thresholds in the adaptive object augmentation paradigm. Hyperparameter t_n represents the maximum number of background point clouds wrapped by the enlarged GT bbox. AOA ($t_n = 0$): no background point clouds.

Method	3D Detection (Car)			BEV Detection (Car)		
	Easy	Mod.	Hard	Easy	Mod.	Hard
Ours w AOA ($t_n = 0$)	89.01	81.29	78.82	92.83	90.37	88.28
Ours w AOA ($t_n = 5$)	**90.15**	**81.66**	**78.97**	**94.12**	**90.71**	**88.35**
Ours w AOA ($t_n = 10$)	89.79	80.98	78.59	92.59	89.98	88.17
Ours w AOA ($t_n = 15$)	88.37	79.62	76.9	92.64	88.88	88.12

optimal performance. This is consistent with our expectation that rectifying the misclassification of pillars within an object is conducive to performance improvements.

We also investigate the effect of voxel size on semantic information representation. As shown in Table 6, appropriately reducing the voxel size can slightly improve the detection performance while increasing the computational cost. As the voxel size is reduced to $v = 0.1^2$, performance begins to decline. This is because the local semantic information of small voxel size is not discriminative. A significant performance decline is observed when the voxel size is gradually increased from 0.16^2. The reason is that larger voxel sizes result in lower detection resolution and may also lead to under-segmentation problems. Therefore, $v = 0.16^2$ is a suitable voxel size for semantic information representation while maintaining efficiency.

4.4.3. Effect of adaptive object augmentation paradigm

Table 7 reports the effectiveness of different thresholds t_n in the AOA paradigm. The integration of AOA ($t_n = 5$) with our approach outperforms that with AOA ($t_n = 0$) by a significant margin. This justifies our attempt to "paste" the augmented object in the ground region containing the appropriate background point cloud, rather than in the completely occluded region (no points). However, the performance is observed to gradually decrease with increasing t_n. This is because objects of "moderate" or "hard" difficulty levels are typically pasted in the sparse point cloud region, and t_n will affect the number of virtual objects in this region, resulting in a sub-optimal performance.

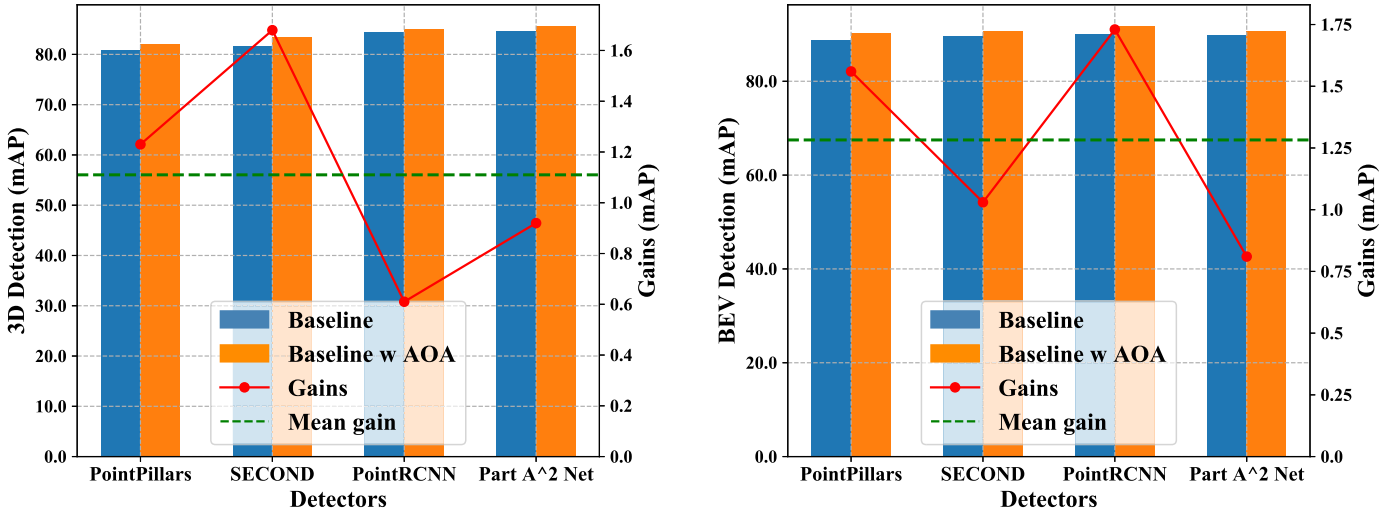

Fig. 10. Effect of embed adaptive object augmentation paradigm into state-of-the-art detectors.

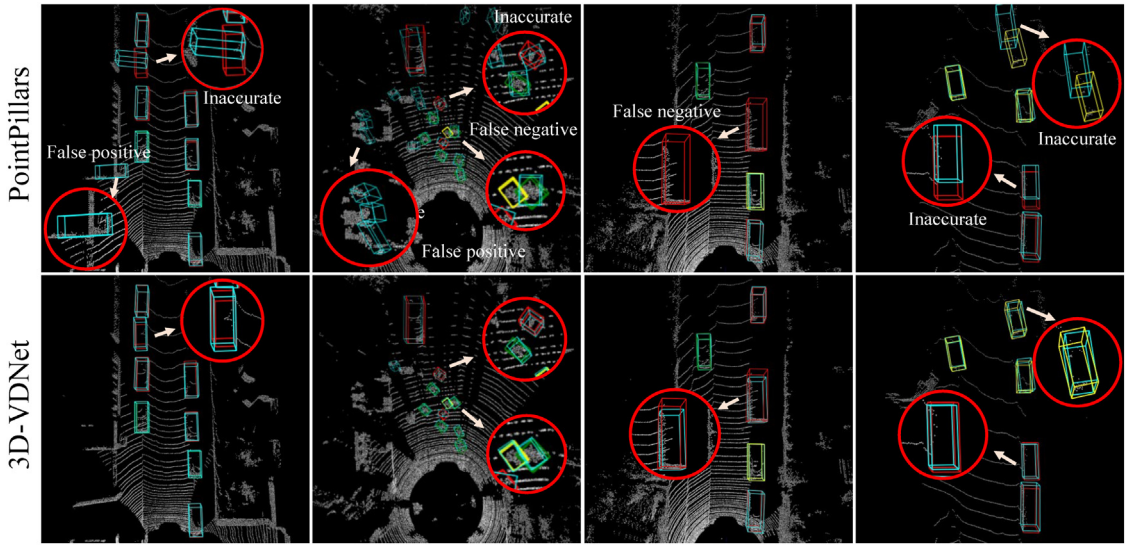

Fig. 11. Qualitative results of the 3D object detection on KITTI *val* split (Top: PointPillars, Bottom: 3D-VDNet). As a reference, we visualize the ground truth objects of easy, moderate, and hard levels as green, yellow, and red, respectively. We also visualize detection results as cyan for comparison.

Table 8
Performance of the proposed method with different configurations for pedestrians and cyclists on the KITTI *val* set. All results are reported by the AP with a 0.5 IoU threshold and 40 recall positions.

	AOA	SFA	SMG	Pedestrian 3D AP			Cyclist 3D AP			3D mAP (Mod.)
				Easy	Mod.	Hard	Easy	Mod.	Hard	
(a)				64.79	58.99	55.6	83.97	60.6	56.33	59.8
(b)	✓			64.96	59.24	54.72	88.34	62.92	58.70	61.08
(c)	✓	✓		66.06	59.54	54.77	89.21	65.72	61.22	62.63
(d)	✓	✓	✓	**67.98**	**61.33**	**57.44**	**88.69**	**65.89**	**61.34**	**63.61**

To further verify the effectiveness of our proposed AOA module, we embed it into state-of-the-art methods, such as SECOND [4], PointRCNN [12], and Part A^2 Net [30]. Fig. 10 demonstrates that following the embedding of our proposed AOA module, all detectors outperform their baseline by a notable margin, with average improvements of 1.1% and 1.3% mAP in 3D and BEV detections, respectively.

4.5. More results on pedestrian and cyclist

To further verify the effectiveness of the proposed modules, extensive experiments are performed on the pedestrian and cyclist classes on the KITTI *val* set. The baseline result for Pedestrian and Cyclist model is achieved by setting the learning rate, training epochs, batch size of each GPU, and the number of GT samples for cyclists as 0.003, 80, 4, 10, respectively. Other implementation details follow PointPillars. The hyperparameters k of the $k \times k$ filter in the SMG module is set 3. Other hyperparameters for the optimal Pedestrian and Cyclist model remain the same as the settings in the Car model. The result in Table 8 demonstrates that our approaches combining AOA, SFA, and SMG modules (d) booms the baseline (a) by a large margin and each proposed module significantly outperforms its counterpart.

5. Conclusion

In the current paper, we explore the potential of employing the point cloud VDCs in pillars to optimize feature extraction and object augmentation. More specifically, by leveraging the VDCs, we design a Spatial Feature Aggregation module to fuse point-interactive and distribution features for the extraction of robust features, construct a voxelized semantic map to spatially enhance semantic embedding and describe a simple yet effective object augmentation paradigm to overcome the conflict between augmented objects and corresponding scenes. Extensive experiments demonstrate that our framework significantly outperforms the baseline in all classes and the proposed object augmentation paradigm that maintains a strong generalization ability under various detectors. Our work provides a new perspective on voxel representations and the potential of the vertical distribution characteristics in point clouds.

CRediT authorship contribution statement

Weiping Xiao: Conceptualization, Methodology, Software, Writing-original-draft. **Xiaomao Li:** Supervision, Validation, Funding-acquisition. **Chang Liu:** Writing-review-editing, Validation. **Jiantao Gao:** Visualization, Data-curation. **Jun Luo:** Funding-acquisition. **Yan Peng:** Project-administration, Funding-acquisition. **Yang Zhou:** Writing-review-editing, Funding-acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the National Key Research and Development Program of China [Grant No. 2020YFC1521700], the Joint Founds of National Natural Science Foundation of China [Grant No. U1813217], and the National Natural Science Foundation of China [Grant No. 51904181].

References

[1] E. Arnold, O.Y. Al-Jarrah, M. Dianati, S. Fallah, D. Oxtoby, A. Mouzakitis, A survey on 3d object detection methods for autonomous driving applications, IEEE Trans. Intell. Transp. Syst. 20 (10) (2019) 3782–3795, https://doi.org/10.1109/Tits.2019.2892405.
[2] P. Hu, J. Ziglar, D. Held, D. Ramanan, What you see is what you get: exploiting visibility for 3d object detection, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 10998–11006. doi:10.1109/CVPR42600.2020.01101.
[3] Y. Zhou, O. Tuzel, Voxelnet: end-to-end learning for point cloud based 3d object detection, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 4490–4499. doi:10.1109/CVPR.2018.00472.
[4] Y. Yan, Y.X. Mao, B. Li, Second: sparsely embedded convolutional detection, Sensors 18 (10) (2018) https://doi.org/10.3390/s18103337.
[5] A.H. Lang, S. Vora, H. Caesar, L.B. Zhou, J.O. Yang, O. Beijbom, PointPillars: fast encoders for object detection from point clouds, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 12689–12697. https://doi.org/10.1109/CVPR.2019.01298.
[6] Y.Y. Ye, H.J. Chen, C. Zhang, X.L. Hao, Z.X. Zhang, SARPNET: shape attention regional proposal network for lidar-based 3d object detection, Neurocomputing 379 (2020) 53–63, https://doi.org/10.1016/j.neucom.2019.09.086.
[7] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, M. Bennamoun, Deep learning for 3d point clouds: a survey, IEEE Trans. Pattern Anal. Mach. Intell. (2020) https://doi.org/10.1109/TPAMI.2020.3005434 1–1.
[8] C.R. Qi, H. Su, K.C. Mo, L.J. Guibas, PointNet: deep learning on point sets for 3d classification and segmentation, in: 2017 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 77–85. https://doi.org/10.1109/CVPR.2017.16.
[9] C.R. Qi, L. Yi, H. Su, L.J. Guibas, PointNet++: deep hierarchical feature learning on point sets in a metric space, in: Advances in Neural Information Processing Systems 30 (NIPS 2017), Vol. 30, 2017, pp. 5099–5108.
[10] C.R. Qi, W. Liu, C.X. Wu, H. Su, L.J. Guibas, Frustum pointnets for 3d object detection from rgb-d data, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 918–927. https://doi.org/10.1109/CVPR.2018.00102.
[11] Z. Yang, Y. Sun, S. Liu, X. Shen, J. Jia, Ipod: Intensive point-based object detector for point cloud, CoRR abs/1812.05276 (2018).
[12] S.S. Shi, X.G. Wang, H.S. Li, Pointrcnn: 3d object proposal generation and detection from point cloud, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 770–779. https://doi.org/10.1109/CVPR.2019.00086.
[13] Z.T. Yang, Y.A. Sun, S. Liu, X.Y. Shen, J.Y. Jia, Std: Sparse-to-dense 3d object detector for point cloud, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 1951–1960. https://doi.org/10.1109/ICCV.2019.00204.
[14] C.R. Qi, O. Litany, K.M. He, L.J. Guibas, Deep hough voting for 3d object detection in point clouds, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 9276–9285. https://doi.org/10.1109/ICCV.2019.00937.
[15] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.Y. Fu, A.C. Berg, SSD: single shot multibox detector, in: Proceedings of the European Conference on Computer Vision (ECCV), vol. 9905, 2016, pp. 21–37. https://doi.org/10.1007/978-3-319-46448-0_2.
[16] S. Shi, C. Guo, L. Jiang, Z. Wang, H. Li, Pv-rcnn: Point-voxel feature set abstraction for 3d object detection, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 10526–10535. doi:10.1109/CVPR42600.2020.01054.
[17] Y.L. Chen, S. Liu, X.Y. Shen, J.Y. Jia, Fast point r-cnn, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 9774–9783. https://doi.org/10.1109/ICCV.2019.00987.
[18] D.L. Wang, I. Posner, Voting for voting in online point cloud object detection, Robot.: Sci. Syst. Xi (2015) https://doi.org/10.15607/RSS.2015.XI.035.
[19] M. Engelcke, D. Rao, D.Z. Wang, C.H. Tong, I. Posner, Vote3deep: fast object detection in 3d point clouds using efficient convolutional neural networks, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 1355–1361. https://doi.org/10.1109/ICRA.2017.7989161.
[20] B. Yang, W.J. Luo, R. Urtasun, Pixor: real-time 3d object detection from point clouds, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 7652–7660. https://doi.org/10.1109/CVPR.2018.00798.
[21] D.-S. Hong, H.-H. Chen, P. Hsiao, L. Fu, S. Siao, Crossfusion net: deep 3d object detection based on rgb images and point clouds in autonomous driving, Image Vis. Comput. 100 (2020), 103955 https://doi.org/10.1016/j.imavis.2020.103955.
[22] S. Song, J. Xiao, Deep sliding shapes for amodal 3d object detection in rgb-d images, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 808–816. https://doi.org/10.1109/CVPR.2016.94.
[23] T.Y. Lin, P. Goyal, R. Girshick, K.M. He, P. Dollar, Focal loss for dense object detection, in: 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2999–3007. https://doi.org/10.1109/ICCV.2017.324.
[24] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the kitti vision benchmark suite, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 3354–3361. https://doi.org/10.1109/CVPR.2012.6248074.

[25] X.Z. Chen, H.M. Ma, J. Wan, B. Li, T. Xia, Multi-view 3d object detection network for autonomous driving, in: 2017 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6526–6534. https://doi.org/10.1109/CVPR.2017.691.

[26] X.Z. Chen, K. Kundu, Y.K. Zhu, A. Berneshawi, H.M. Ma, S. Fidler, R. Urtasun, 3d object proposals for accurate object class detection, in: Advances in Neural Information Processing Systems 28 (NIPS 2015), vol. 28, 2015, pp. 424–432.

[27] D. Kingma, J. Ba, Adam: a method for stochastic optimization, in: 3rd International Conference on Learning Representations ICLR, 2015.

[28] L.N. Smith, N. Topin, Super-convergence: very fast training of neural networks using large learning rates, Artif. Intell. Mach. Learn. Multi-Domain Oper. Appl. 11006 (2019) https://doi.org/10.1117/12.2520589.

[29] S. Zhang, C. Chi, Y. Yao, Z. Lei, S.Z. Li, Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition CVPR, 2020, pp. 9756–9765. https://doi.org/10.1109/CVPR42600.2020.00978.

[30] S. Shi, Z. Wang, J. Shi, X. Wang, 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. (2020) https://doi.org/10.1109/TPAMI.2020.2977026 1–1.

[31] M. Liang, B. Yang, Y. Chen, R. Hu, R. Urtasun, Multi-task multi-sensor fusion for 3d object detection, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 7337–7345. https://doi.org/10.1109/CVPR.2019.00752.

[32] J. Ku, M. Mozifian, J. Lee, A. Harakeh, S.L. Waslander, Joint 3d proposal generation and object detection from view aggregation, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018, pp. 5750–5757. https://doi.org/10.1109/IROS.2018.8594049.

[33] Z.X. Wang, K. Jia, Frustum convnet: sliding frustums to aggregate local point-wise features for amodal 3d object detection, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019, pp. 1742–1749. https://doi.org/10.1109/IROS40897.2019.8968513.

[34] M. Simon, S. Milz, K. Amende, H.M. Gross, Complex-yolo: an euler-region-proposal for real-time 3d object detection on point clouds, in: Proceedings of the European Conference on Computer Vision (ECCV), vol. 11129, 2019, pp. 197–209. https://doi.org/10.1007/978-3-030-11009-3_11.

[35] M. Liang, B. Yang, S.L. Wang, R. Urtasun, Deep continuous fusion for multi-sensor 3d object detection, in: Proceedings of the European Conference on Computer Vision (ECCV), vol. 11220, 2018, pp. 663–678. https://doi.org/10.1007/978-3-030-01270-0_39.