RBGNet: Ray-based Grouping for 3D Object Detection

Haiyang Wang[1] Shaoshuai Shi[2*] Ze Yang[3] Rongyao Fang[4]
Qi Qian[5] Hongsheng Li[4] Bernt Schiele[2] Liwei Wang[6,7]
[1]Center for Data Science, Peking University [2]Max Planck Institute for Informatics
[3]University of Toronto [4]The Chinese University of Hong Kong [5]Alibaba Group
[6]Key Laboratory of Machine Perception, MOE, School of Artificial Intelligence, Peking University
[7]International Center for Machine Learning Research, Peking University
{wanghaiyang@stu, wanglw@cis}.pku.edu.cn {sshi, schiele}@mpi-inf.mpg.de
zeyang@cs.toronto.edu {rongyaofang@link, hsli@ee}.cuhk.edu.hk qi.qian@alibaba-inc.com

Abstract

As a fundamental problem in computer vision, 3D object detection is experiencing rapid growth. To extract the point-wise features from the irregularly and sparsely distributed points, previous methods usually take a feature grouping module to aggregate the point features to an object candidate. However, these methods have not yet leveraged the surface geometry of foreground objects to enhance grouping and 3D box generation. In this paper, we propose the RBGNet framework, a voting-based 3D detector for accurate 3D object detection from point clouds. In order to learn better representations of object shape to enhance cluster features for predicting 3D boxes, we propose a ray-based feature grouping module, which aggregates the point-wise features on object surfaces using a group of determined rays uniformly emitted from cluster centers. Considering the fact that foreground points are more meaningful for box estimation, we design a novel foreground biased sampling strategy in downsample process to sample more points on object surfaces and further boost the detection performance. Our model achieves state-of-the-art 3D detection performance on ScanNet V2 and SUN RGB-D with remarkable performance gains. Code will be available at https://github.com/Haiyang-W/RBGNet.

1. Introduction

3D object detection is becoming an active research topic in computer vision, which aims to estimate oriented 3D bounding boxes and semantic labels of objects in 3D scenes. As a fundamental technique for 3D scene understanding, it plays a critical role in many applications, such as autonomous driving [3, 38], augmented reality [2, 4] and domestic robots [54, 39]. Unlike the scenarios in the well-studied 2D image problems, 3D scenes are generally represented by point clouds, a set of unordered, sparse and irregular points captured by depth sensors (*e.g.*, RGB-D cameras, LiDAR sensors), which makes it significantly different from traditional regular input data like images and videos.

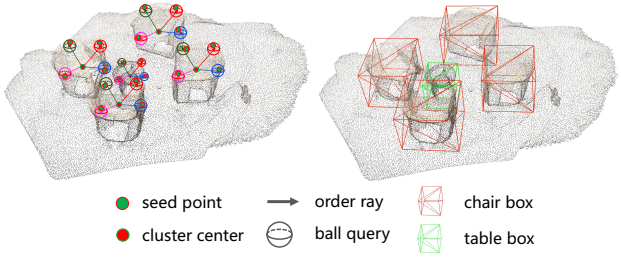

Figure 1: **3D object detection in point clouds with a ray-based feature grouping module.** Given the point clouds of a 3D scene, our model aggregates the point-wise features on object surface by a group of ordered rays to boost the performance of 3D object detection.

Previous 3D detection approaches can be coarsely classified into two lines in terms of point representations, *i.e.*, the grid-based methods and the point-based methods. The grid-based methods generally convert the irregular points to regular data structure such as 3D voxels [37, 35, 44, 53, 32, 48, 9] or 2D bird's eye view maps [6, 16, 19, 45, 46]. Thanks to the great success of PointNet series [30, 31], the point-based methods [29, 34, 28, 49, 20, 7] directly extract the point-wise features from the irregular and points. These point-wise features are generally enhanced by various feature grouping modules for predicting the 3D bounding boxes. However, these feature grouping strategies have not well explored the fine-grained surface geometry to help improve the performance of 3D box generation.

We argue that feature grouping module plays an important role in point-based 3D detectors, and how to better incorporate the foreground object geometry features to en-

*Corresponding author: Shaoshuai Shi.

GT-Features (explicit GT-center)	GT-Features (implicit GT-center)	FgSamp	mAP@0.25	mAP@0.50
			62.90	39.91
✓			76.21 (+13.31)	53.91 (+14.00)
	✓		71.69 (+8.79)	50.71 (+10.80)
		✓	71.27 (+8.39)	50.68 (+10.77)

Table 1: Results of VoteNet [28] variants on ScannetV2 [8]. **GT-Features**: Aggregate the features of ground-truth surface points for the 3D box generation of this object, where the "explicit GT-center" / "implicit GT-center" indicate that the above features are grouped to the ground-truth center / predicted vote centers, respectively. **FgSamp:** FPS is only conducted on foreground points.

hance the quality of point-wise features is the key to predict better 3D bounding boxes. As shown in Table 1, for the popular VoteNet [28] point-wise 3D detector, by simply grouping the features of accurate object surface points to the features of their correct vote centers, the performance can be improved dramatically with a gain of 13.31 on mAP@0.25 for explicit usage of ground truth labels (2^{nd} row of Table 1), and a gain of 8.79 for implicit usage of ground truth labels (3^{rd} row of Table 1). Here the "explicit usage" indicates that the ground truth labels are not only utilized for grouping the object surface points but also for replacing the vote centers with ground truth centers, while the "implicate usage" means the ground truth labels are only used for grouping the object surface points. These facts inspire us to explore on designing a better feature representation for the surface geometry of foreground objects, to help the prediction of 3D bounding boxes.

Hence, we present a new 3D detection framework, RBGNet, which is a one-stage 3D detector for 3D object detection from raw point clouds. Our RBGNet is built on top of VoteNet [28], and we propose two novel strategies to boost the performance of 3D object detection by implicitly learning from foreground object features.

Firstly, we propose the *ray-based feature grouping* that could learn better feature representation of the surface geometry of foreground objects. The learned features are utilized to augment the cluster features for 3D boxes estimation. Specifically, we formulate a ray-based mechanism to capture the object surface points, where a number of rays are uniformly emitted from the cluster center with the determined angles (see Fig. 1). The far bounds of the rays are based on our predicted object scale of this cluster. Then a number of anchor points is densely sampled on each ray, where the aggregated local features of each anchor point are utilized to predict whether they are on the object surface to learn the geometry shape. Moreover, a coarse-to-fine strategy is proposed to generate different number of anchor points based on the sparsity of different regions. The learned features from all the anchor points will be finally aggregated to boost the features of cluster centers for predicting 3D bounding boxes. The experiments (Table 4) show that our ray-based feature grouping strategy can effectively encode the surface geometry of foreground objects and significantly improves 3D detection performance.

Secondly, we propose the *foreground biased sampling* strategy to allocate more foreground object points for predicting 3D boxes. We observe that the points on object surfaces are more useful than those on the background for 3D box estimation (similar observations are also mentioned by [47, 32]), and 4^{th} row of Table 1 shows that by conducting farthest point sampling only on the ground truth foreground points, the performance of VoteNet [28] could be boosted from 62.90 to 71.27 in terms of mAP@0.25. Therefore, we propose a simple but effective strategy to sample points biased towards object surface while still keeping the coverage rate of the whole scene. Specifically, we append a segmentation head to the point-wise features before each farthest point sampling, where the head will predict the confidence of each point being a foreground point. According to the ranking of their foreground scores, the input points are separated into foreground set and background set. And these two sets will apply farthest point sampling separately, where we sample most target points (*i.e.*, 87.5% in our case) from the foreground set and a small number (*i.e.*, 12.5%) from the background set to keep the coverage rate of the whole scene. Our foreground biased sampling can produce a more informative sampling of points over foreground objects surface for feature extraction, and the performance gains (Table 4) demonstrate its effectiveness.

In a nutshell, our contributions are three-fold: 1) We propose a novel ray-based feature grouping module to encode object surface points with determined rays, which can learn better surface geometry features of objects to boost the performance of point-based 3D object detectors. 2) We present foreground biased sampling module to focus feature learning of the network on foreground surface points while also keeping the coverage rate for the whole scene, which can incorporate more object points to benefit point-based 3D box generation. 3) Equipped with the above two modules, our proposed RBGNet framework outperforms state-of-the-art methods with remarkable margins both on ScanNetV2 [8] and SUN RGB-D [36].

2. Related Work

3D Object Detection is challenging due to the irregular, sparse and orderless characteristics of 3D points. Most existing works could be classified into two categories in terms of point cloud representations, *i.e.*, grid-based and point-based. Grid-based approaches transform point clouds to regular data, such as 2D grids [6, 53, 46] or 3D voxels [35, 44, 15, 32, 51, 52, 48, 33]. 2D grid methods project point clouds to a bird view before proceeding to the rest of the pipeline. Voxel-based methods convert the point clouds into 3D voxels to be processed by 3D CNN or efficient 3D sparse convolution [12], which greatly facilitate 3D object detection. Popularized by PointNet [30] and

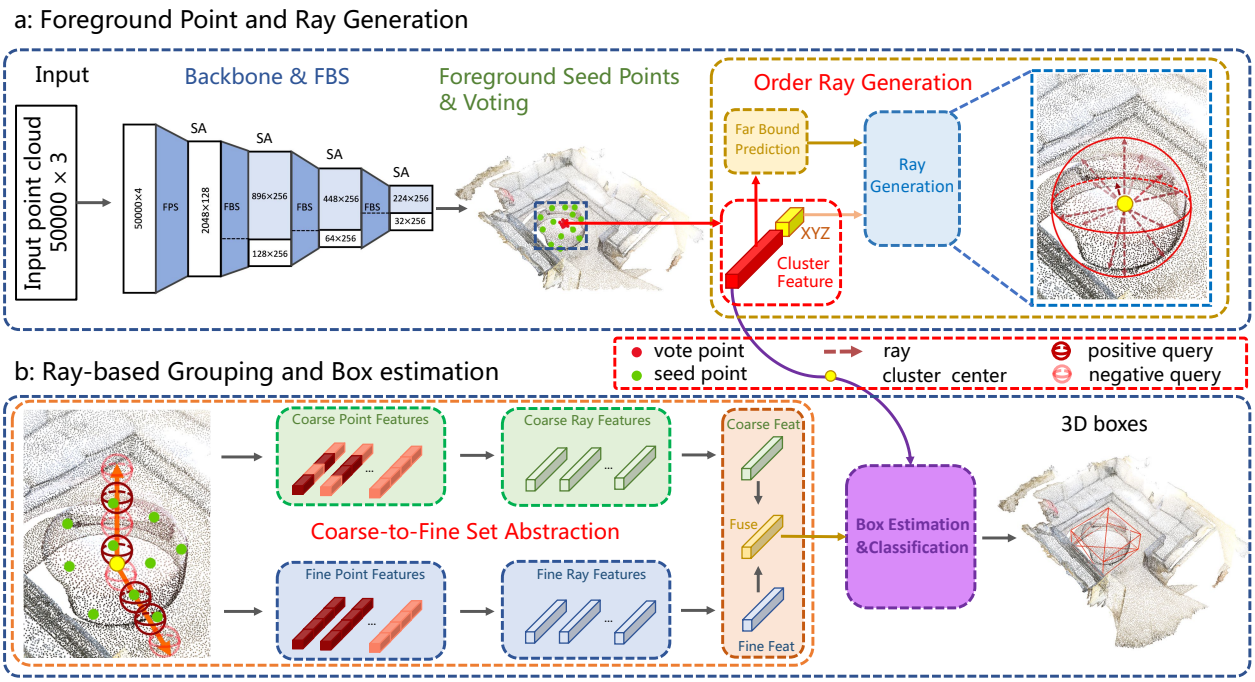

Figure 2: The RBGNet architecture for 3D object detection from point cloud. (a) Generating more foreground seed points and a number of rays emitted from object centers. (b) Object shape encoding by ordered rays and 3D bounding box estimation.

its variants [31, 14, 50], point-based methods [28, 49, 20] have become extensively employed on estimating bounding box directly from raw points. Most of existing methods can be considered as a bottom-up manner, which requires point grouping step to obtain object features. Point R-CNN [34] groups point features within the 3D proposals by point cloud region pooling. VoteNet [28] applies Hough Voting to group the points that vote to the similar center region. Group-free [20] implicitly groups point features by an attention module. Although these methods have explored various feature grouping strategies, they have not leveraged the surface geometry of foreground objects. We propose a novel ray-based feature grouping module to encode object shape distribution with determined rays, and the learned features are used to further boost 3D detection performance.

2D Shape Representation. Shape representation [42, 25, 43, 1, 18, 26] is of particular interest due to the ability to explicitly describe the 2D object shape with points. Polar Mask [40] and ESE-SEG [43] both use polar representation to model the object boundary and then regress the object locations as well as the length of rays emitting uniformly from the object centroids. However, these shape representations may fail to model 3D object surface, because of the limited expressive ability on concave shape, the difficulty of inner center definition. We design a ray-based 3D shape representation to effectively model object surface geometry.

Point Cloud Sampling. Sampling [23, 17, 10] aims to represent the original point cloud in a sparse way, plays a key role in point cloud analysis. Farthest point sampling (FPS) has been widely used as a pooling operation [31, 28], since it can uniformly sample distributed points. However, FPS is agnostic to downstream tasks by a predefined rule, foreground instances with few interior points may lose all points after sampling. 3DSSD [47] applies a fused FPS based on feature and euclidean distance, but still does not focus on foreground points explicitly. To deal with the dilemma, we design a simple but effective strategy, *foreground biased sampling*, to sample more points on object surface while still keeping the coverage rate of the whole scene.

3. Methodology

This section describes the technical details of the proposed RBGNet detector. §3.1 briefly presents the overview of our approach. Next, §3.2 to §3.4 elaborate on the network design and the learning objective.

3.1. Overview

RBGNet is a one-stage 3D object detection framework aiming at more accurate bounding box estimation from irregular point clouds. As illustrated in Fig. 2, RBGNet consists of three major components: i) a *backbone network* with foreground biased sampling to extract feature representation from point clouds, ii) a *ray-based feature grouping* module to effectively capture the points on object surface and learn from the shape distribution to augment cluster feature and iii) a *proposal and classification module* followed by 3D non-maximum-suppression (NMS). Our paper mainly focuses on the sampling and grouping modules, so we follow the same proposal and classification strategy as

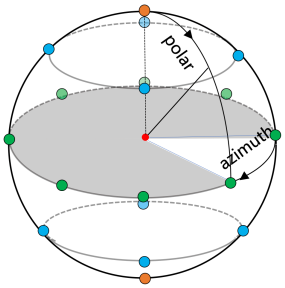

Figure 3: Demonstration of spherical coordinate system and the distribution of 18 rays.

in VoteNet [28] to estimate final bounding boxes. We will describe the technical details in the following parts.

3.2. Ray-based Feature Grouping

VoteNet [28] has shown tremendous success for 3D object detection. After getting the seed points from the backbone (PointNet++ [31]), it reformulates traditional Hough voting, and generates object candidates by grouping the seed points whose votes are within the same cluster. The aggregated feature is then used to estimate the 3D bounding boxes and associated semantic labels. However, the quality of the grouping principally determines the reliability of proposal features and detector performance. Some follow-up works [5, 7, 42] are actually trying to solve this problem, but they have not well explored on the fine-grained surface geometry of foreground objects. To address this limitation, we propose the *ray-based feature grouping* module, which can effectively encode the shape distribution and learn better object features to enhance 3D detection performance.

3.2.1 Ray Point Representation.

We first illustrate the process of our proposed ray point representation, where two types of anchor points are generated on each ray to encode the object geometry around the cluster centers. These anchor points of all rays will be utilized for the final feature enhancement in §3.2.2.

Formulation of determined rays. We generate a set of vote cluster centers $\{c_i\}_{i=1}^{M}$ based on the vote sampling and grouping defined in VoteNet [28], where $c_i = [v_i, f_i]$ indicating the vote center $v_i \in \mathbb{R}^3$ and its features $f_i \in \mathbb{R}^C$, M is the number of vote clusters. For each vote cluster, N rays are emitted uniformly from the cluster center with the determined angles and far bounds. As shown in Fig. 3, the rules for generating rays are based on the spherical coordinate system, which are formulated as follows:

- The polar angle $\theta \in [0, \pi]$ is split into P bins, and each bin corresponds a round surface that is perpendicular to the Z-axis. The angle of p^{th} bin is:

$$\theta_p = \frac{\pi p}{(P\text{-}1)}, \quad p \in \{0, ..., P\text{-}1\}. \tag{1}$$

- The number of rays (denoted as A_p) terminated on the p^{th} round surface is calculated as follows:

$$A_p = \begin{cases} 1, & \text{if } p = 0 \text{ or } P\text{-}1, \\ 4 \times p, & \text{if } 0 < p \leq \dfrac{P\text{-}1}{2}, \\ 4 \times (P\text{-}p\text{-}1), & \text{if } P\text{-}1 > p > \dfrac{P\text{-}1}{2}, \end{cases} \tag{2}$$

  where 4 is a hyper-parameter to indicate the factor for the number of sampled rays in each round surface.
- With A_p and θ_p, a ray could be determined. Its azimuth angle $\psi_{p,a} \in [0, 2\pi]$ and polar angle $\theta_{p,a} \in [0, \pi]$ could be formulated as follows:

$$\psi_{p,a} = \frac{2\pi a}{A_p}, \quad \theta_{p,a} = \theta_p, \quad a \in \{0, ...A_p\text{-}1\}. \tag{3}$$

Our adopted strategy could generate more uniformly distributed rays to better cover the surrounding region of clusters. Given the polar-bin number P, the number of rays is $N = \sum_{p=0}^{P-1} A_p$ (*i.e.* $P = 9 \rightarrow N = 66$ in our case). Note that more rays will be generated when the polar angle is closer to $\frac{\pi}{2}$.

As for the far bounds of the rays of each cluster, all the rays are of the same length as the object scale l_i, which is predicted based on the cluster features f_i. Here, we explicitly supervise the object scale l_i by regression loss

$$L_{\text{scale-reg}} = \frac{1}{I} \sum_i ||l_i - l_i^*||_\eta \, \mathbb{I}[i_{th} \text{ is positive}], \tag{4}$$

where l_i^* is the half diagonal side of the assigned GT box and $\mathbb{I}[i_{th}$ is positive$]$ is the indicator function to indicate whether the vote center c_i is around a GT object center (within a radius of $0.3m$). I is the number of positive vote centers. η means smooth-ℓ_1 norm.

Coarse-to-fine anchor point generation. After generating determined rays, RBGNet samples a number of anchor points along each ray. However, it is inefficient to directly sampling points: 1) the less important free space and background region are still sampled, 2) the number of sampled points in a ball query operation is limited, and thus a lot of object points in dense areas are not captured. To address these limitations, we propose a coarse-to-fine anchor point generation strategy. It increases the sample efficiency by querying more points for dense areas. Inspired by the success of [21], we adopt a hybrid sampling strategy, which contains two sampling processes: one "coarse" and one "fine". Before generating anchor points, we first up-sample the seed points back to 2048 points by trilinear interpolation to obtain more meaningful points, especially on object surface. The target point positions for upsampling are the same as the 1^{st} SA layers of PointNet++ [31] backbone. For the i^{th} cluster center with cluster features f (we remove the subscript i of f_i for simplicity), we conduct the coarse-to-fine anchor point generation in the following process.

Firstly, in coarse stage, as for the n^{th} ray, we sample a set of anchor points as

$$Q_n^{(c)} = \{q_{n,k}^{(c)} = (x_{n,k}^{(c)}, y_{n,k}^{(c)}, z_{n,k}^{(c)})\}, k \in \{1, \cdots, K_c\}, \quad (5)$$

where K_c is the number of anchor points sampled on each ray, and the anchor points are generated by stratified sampling to evenly partition the ray into K_c bins.

To extract local feature of each anchor point, we apply set abstraction [31] to aggregate the features of the seed points around each anchor point. The aggregated local features of anchor point $q_{n,k}^{(c)}$ is denoted as $\rho_{n,k}^{(c)}$. Finally, we append a binary classification module for estimating the positive mask $m_{n,k}^{(c)}$ of point $q_{n,k}^{(c)}$ based on cluster feature f and local feature $\rho_{n,k}^{(c)}$ as follows:

$$m_{n,k}^{(c)} = \mathcal{F}_{mask}^{(c)}(\rho_{n,k}^{(c)}, f), \quad (6)$$

where the ground-truth of positive masks are calculated by applying ball query operation [31] for each anchor point. We assign positive label to an anchor point if some surface points of its assigned GT object are within its ball query region, or the anchor point will be assigned with a negative label. Hence this point mask module could predict whether each anchor point belongs to its corresponding object or not.

Secondly, in fine stage, different from [21] which computes sample probability from point density, our fine anchor points are biased towards the dense part of its corresponding object. To achieve this goal, we apply inverse transform sampling to uniformly generate some K_f anchor points set $Q_n^{(f)} = \{q_{n,k}^{(f)}\}_{k=1}^{K_f}$ on positive regions (predicted by the point mask module of coarse stage) of each ray. As adopted in the coarse branch, we also extract the local features and predict the positive mask for each fine anchor point.

Repeat this coarse-to-fine process on all rays, we obtain the coarse and fine local point feature set $\mathcal{P}^{(c)} = \{\rho_{n,k}^{(c)}\}_{k=1,n=1}^{K_c,N}$, $\mathcal{P}^{(f)} = \{\rho_{n,k}^{(f)}\}_{k=1,n=1}^{K_f,N}$, point mask set $\mathcal{M}^{(c)} = \{m_{n,k}^{(c)}\}_{k=1,n=1}^{K_c,N}$, $\mathcal{M}^{(f)} = \{m_{n,k}^{(f)}\}_{k=1,n=1}^{K_f,N}$ and their corresponding positions of anchor points.

3.2.2 Feature Enhancement by Determined Rays

As discussed in §1, the fined-grained surface geometry of foreground objects plays a crucial role in generating accurate object proposals. The process of our proposed coarse-to-fine anchor point generation already encodes such a surface geometry features, since our predicted surface masks and learned local features could implicitly describe the object geometry. Here we propose to aggregate those informative features of the anchor points to enhance the quality of cluster features, where the order of rays plays an important roles in the feature aggregation. .

To be specific, given the local features $\mathcal{P}^{(c)}$ and $\mathcal{P}^{(f)}$, the point masks $\mathcal{M}^{(c)}$ and $\mathcal{M}^{(f)}$ of each anchor point, the local features of each anchor points will be masked by setting the features of negative anchor points to zeros. We denote the masked features as $\hat{\mathcal{P}}^{(c)}$ and $\hat{\mathcal{P}}^{(f)}$.

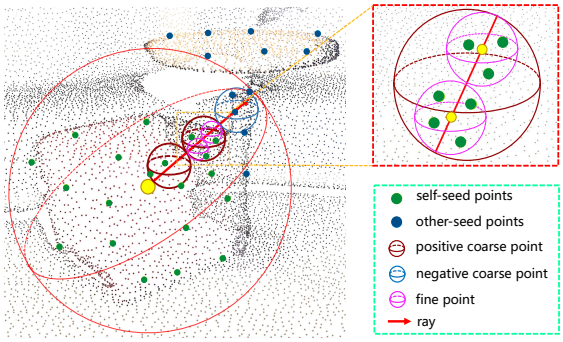

Figure 4: Illustration of coarse-to-fine anchor point generation. Fine sampling is biased towards the dense part of its corresponding object.

To aggregate the learned features orderly based on the determined rays, we formulate a fusion stage to integrate point features in a predefined order of rays. The features of coarse and fine anchor points are aggregated with two separate branches. In coarse branch, the masked point features of n^{th} ray, $\{\hat{\rho}_{n,k}^{(c)}\}_{k=1}^{K_c}$, are firstly fused into a single ray feature $r_n^{(c)}$. It is implemented by concatenating the features of anchor points in order before being projected to a 32-dimensional features:

$$r_n^{(c)} = \mathcal{F}_{\text{point}}^{(c)}(\{\hat{\rho}_{n,k}^{(c)}\}_{k=1}^{K_c}, \odot), \quad (7)$$

where $\odot$ means the concatenation operation. Then, in the same way, we concatenate all the ray features $\mathcal{R}^{(c)} = \{r_n^{(c)}\}_{n=1}^{N}$ with a determined order and apply a two-layer MLP to generate a 128-dimensional coarse feature:

$$\mu^{(c)} = \mathcal{F}_{\text{ray}}^{(c)}(\{r_n^{(c)}\}_{n=1}^{N}, \odot). \quad (8)$$

Note that the predefined order of both anchor points and rays are consistent for each proposal, but different ordering strategies do not affect the performance. The fine branch also adopts the same strategy as the coarse branch to generates a 128-dimensional feature $\mu^{(f)}$.

Finally, the coarse and fine features are fused as:

$$g = \mathcal{F}_{\text{fuse}}(\mu^{(c)}, \mu^{(f)}). \quad (9)$$

The fused feature g is finally combined with the cluster feature f to improve the performance of 3D object detection.

In this way, our RBGNet models the surface geometry implicitly and roughly obtain the size and the position of a possible object in a class-agnostic way, which could greatly benefit the prediction of 3D bounding boxes.

3.3. Foreground Biased Sampling

The foreground points provide rich information on predicting their associated object locations and orientations,

and force network to capture shape information for more accurate 3D box generation. However, the widely-adopted farthest point sampling algorithm in the backbone is agnostic to the downstream tasks and samples a lot of background points. It may bring negative effects for 3D detection. Therefore, we design a simple but effective strategy, *Foreground Biased Sampling*, to sample more points on foreground object surfaces while still keeping the coverage rate of the whole scene.

Given the point-wise features encoded by each set abstraction layer, we append a segmentation head for estimating the confidence of each points. The ground-truth segmentation mask is naturally provided by the 3D ground-truth boxes. To be specific, for example, after going through the first SA layer of standard PointNet++ [31], we obtain 2048 downsample point set $\mathcal{D} = \{d_j\}_{j=1}^{2048}$ with xyz ϱ_j and 128-dimensional feature ν_j. Then the segmentation head scores each point to be a foreground point or not as:

$$\varepsilon_j = \mathcal{F}^{\text{fore}}(\nu_j, \varrho_j) \in [0, 1]. \tag{10}$$

We sort the confidence scores, select top κ to form a foreground set $\mathcal{D}^{(f)} = \{d_j^{(f)}\}_{j=1}^{\kappa}$ and the rest are the background set $\mathcal{D}^{(b)} = \{d_j^{(b)}\}_{j=1}^{2048\text{-}\kappa}$. Due to the concentration of high score points, there is a trade-off between the recall of foreground points and the sampling coverage for the whole scene. Based on this observation, we apply farthest point sampling on foreground and background set separately, and combine them into the final sample set as follows:

$$\mathcal{D}^{(\hat{f})} = \text{FPS}(\mathcal{D}^{(f)}), \mathcal{D}^{(\hat{b})} = \text{FPS}(\mathcal{D}^{(b)}), \mathcal{S} = \mathcal{D}^{(\hat{f})} \oplus \mathcal{D}^{(\hat{b})} \tag{11}$$

where $\mathcal{D}^{(\hat{f})} = \{d_j^{(\hat{f})}\}_{j=1}^{\alpha}$ and $\mathcal{D}^{(\hat{b})} = \{d_j^{(\hat{b})}\}_{j=1}^{\beta}$, α and β are the sample number of foreground and background set. $\mathcal{S}$ is the set of final sampled points, which contains more object points while still keeping the coverage rate of the whole scene. In our case, we sample most target points,(*i.e.*, 87.5%) from the foreground set and a small number (*i.e.*, 12.5%) from background set. For example, in downsample process of the 2^{th} SA layer (2048 $\rightarrow$ 1024), κ, α and β are 1024, 896 and 128, respectively.

We adopt the cross entropy loss for foreground segmentation. In inference, the confidence score is obtained by the margin between positive class and negative class.

3.4. Learning Objective

The loss function consists of foreground biased sampling $\mathcal{L}_{\text{fbs}}$, voting regression $\mathcal{L}_{\text{vote-reg}}$, ray-based feature grouping $\mathcal{L}_{\text{rbfg}}$, objectness $\mathcal{L}_{\text{obj-cls}}$, bounding box estimation $\mathcal{L}_{\text{box}}$, and semantic classification $\mathcal{L}_{\text{sem-cls}}$ losses.

$$\begin{aligned} L = \lambda_{\text{vote-reg}}\mathcal{L}_{\text{vote-reg}} + \lambda_{\text{fbs}}\mathcal{L}_{\text{fbs}} + \lambda_{\text{rbfg}}\mathcal{L}_{\text{rbfg}} + \\ \lambda_{\text{obj-cls}}\mathcal{L}_{\text{obj-cls}} + \lambda_{\text{box}}\mathcal{L}_{\text{box}} + \lambda_{\text{sem-cls}}\mathcal{L}_{\text{sem-cls}}. \end{aligned} \tag{12}$$

[1]We report the results of MMDetection3D (https://github.com/open-mmlab/mmdetection3d), which are better than the official paper.

Following the setting in VoteNet [28], we use the same label assignment and loss terms $\mathcal{L}_{\text{vote-reg}}$, $\mathcal{L}_{\text{obj-cls}}$, $\mathcal{L}_{\text{box}}$ and $\mathcal{L}_{\text{sem-cls}}$. $\mathcal{L}_{\text{fbs}}$ is a cross entropy loss used to supervise foreground sampling (see §3.3). $\mathcal{L}_{\text{rbfg}}$ is the sum loss of ray-based feature grouping module defined as follows:

$$\mathcal{L}_{\text{rbfg}} = \lambda_{\text{scale-reg}}\mathcal{L}_{\text{scale-reg}} + \lambda_{\text{c-cls}}\mathcal{L}_{\text{c-cls}} + \lambda_{\text{f-cls}}\mathcal{L}_{\text{f-cls}}. \tag{13}$$

As defined in §3.2, $\mathcal{L}_{\text{scale-reg}}$ is a smooth ℓ_1 loss, to explicitly supervise object scale of each proposal. $\mathcal{L}_{\text{c-cls}}$ and $\mathcal{L}_{\text{f-cls}}$ are both cross entropy losses, to supervise our model for querying valid point on each object surface. The detailed balancing factors are in Appendix.

4. Experiments

4.1. Datasets and Evaluation Metric

We evaluate our method on two large-scale indoor 3D scene datasets, *i.e.*, ScanNet V2 [8] and SUN RGB-D [36], and we follow the standard data splits [28] for both of them.

SUN RGB-D [36] is a single-view RGB-D dataset for 3D scene understanding, which consists of 5K RGB-D training images annotated with the oriented 3D bounding boxes and the semantic labels for 10 categories. Following the standard data processing in [28], we convert the depth images to point clouds using the provided camera parameters.

ScanNet V2 [8] consists of richly-annotated 3D reconstructions of indoor scenes. It consists of 1513 training samples (reconstructed meshes converted to point clouds) with axis-aligned bounding box labels for 18 object categories. Compared to SUN RGB-D, its scenes are larger and more complete with more objects. We sample point clouds from the reconstructed meshes by following [28].

For both datasets, the evaluation follows the same protocol as in VoteNet [28] using mean average precision(mAP) under different IoU thresholds, *i.e.*, 0.25 and 0.5.

4.2. Implementation Details.

Network Architecture Details. For each 3D scene in the training set, we subsample 50000 points from the scene point cloud as the inputs. For the backbone and voting layers, we follow the same network structure of [28], but replace FPS with our proposed Foreground Biased Sampling (FBS) in $2^{nd} \rightarrow 4^{th}$ SA layers. More network details about other parts are given in Appendix.

Training Scheme. Our network is end-to-end optimized by using the AdamW optimizer with the batch size 8 per-GPU and initial learning rate of 0.006 for ScanNet V2 and 0.004 for SUN RGB-D. We train the network for 360 epochs on both datasets, and the initial learning rate is decayed by 10x at the 240-th epoch and the 330-th epoch. The gradnorm clip is applied to stabilize the training dynamics.

ScanNet V2	Backbone	mAP@0.25	mAP@0.5
F-PointNet [29]*	PointNet	19.8	10.8
3D-SIS [13]*	3D ResNet	40.2	22.5
HGNet [5]	GU-net	61.3	34.4
VoteNet [28][1]	PointNet++	62.9	39.9
3D-MPA [11]	MinkNet	64.2	49.2
H3DNet [49]	PointNet++	64.4	43.4
3Detr [22]	PointNet++	65.0	47.0
BRNet [7]	PointNet++	66.1	50.9
VENet [41]	PointNet++	67.7	-
Group-free [20]	PointNet++	67.2(66.6)	49.7(49.0)
Our(R66, O256)	PointNet++	**70.2(69.6)**	**54.2(53.6)**
H3DNet [49]	4×PointNet++	67.2	48.1
Group-free [20]	PointNet++w2×	69.1(68.6)	52.8(51.8)
Our(R66, O512)	PointNet++w2×	**70.6(69.9)**	**55.2(54.7)**

Table 2: Performance comparison on the ScanNetV2 [8] val set with state-of-the-art. The main comparison is based on the best results of multiple experiments, the number within the bracket is the average result of 25 trials. Note that * means it uses RGB as addition inputs.

SUN RGB-D	Backbone	mAP@0.25	mAP@0.5
F-PointNet [29]*	PointNet	54.0	-
ImVoteNet [27]*	PointNet++	63.4	-
MTC-RCNN [24]*	PointNet++	65.3 (64.7)	48.6 (48.2)
3Detr [22]	PointNet++	59.1	32.7
VoteNet [28][1]	PointNet++	59.1	35.8
MLCVNet [42]	PointNet++	59.8	-
HGNet [5]	GU-net	61.6	-
H3DNet [49]	4×PointNet++	60.1	39.0
BRNet [7]	PointNet++	61.1	43.7
VENet [41]	PointNet++	62.5	39.2
Group-free [20]	PointNet++	63.0(62.6)	45.2(44.4)
Our(R66, O256)	PointNet++	**64.1(63.6)**	**47.2(46.3)**

Table 3: 3D object detection results on the SUN-RGB-D [36] val set. The main comparison is also based on the best results of multiple experiments between different methods. Note that * means it uses RGB as addition inputs and our method is geometric only.

4.3. Comparison with state-of-the-art methods.

For performance benchmarking, we compare with a wide range state-of-the-art methods on ScanNet V2 and SUN RGB-D. We follow the previous work [20] and also report both best results and average results.

ScanNet V2. The results are summarized in Table 2. With the same backbone network of a standard PointNet++, our approach achieves 70.2 mAP@0.25 and 54.2 mAP@0.5 using 66 rays and 256 object candidates, which is 2.5 and 3.3 better than previous best methods [41, 7] using the same backbones. With stronger backbones and more sampled object candidates just like [20], *i.e.*, 2× more channels and 512 candidates, our approach is also improved dramatically, achieving 70.6 mAP@0.25 and 55.2 mAP@0.5, which is still 1.5 and 2.4 better than [20]. Notably, we also only use geometric input (point cloud) as previous works did.

FBS	Ray Feat (66)	mAP@0.25	mAP@0.5
		66.2	48.2
✓		67.1	49.0
	✓	69.0	52.9
✓	✓	**69.6**	**53.6**

Table 4: Effect of ray-based feature grouping module and foreground biased sampling.

number of ray	objp recall	mAP@0.25	mAP@0.5
0	-	67.1	49.0
6	38.1	68.4	51.6
18	63.3	68.7	52.0
38	75.8	69.2	52.7
66	78.1	69.6	53.6
102	86.5	**69.9**	**53.9**

Table 5: Ablation study on the performance of ray-base feature grouping module with different ray number.

method	mAP@0.25	mAP@0.5
Voting [28]	67.1	49.0
RoI-Pooling [34]	67.6	49.9
Back-tracing [7]	67.7	50.1
Group-free [20]	68.1	50.5
Our(R66)	**69.6**	**53.6**

Table 6: Comparison with other grouping-based methods.

method	dataset	2^{nd}(1024)	3^{rd}(512)	4^{th}(256)
FPS [28]	ScanNet V2	31.1	30.8	30.3
F-FPS [47]	ScanNet V2	40.4	42.1	43.8
Ours	ScanNet V2	**51.2**	**73.2**	**87.8**
FPS [28]	SUN RGB-D	17.9	17.8	17.7
F-FPS [47]	SUN RGB-D	21.3	22.9	23.5
Ours	SUN RGB-D	**30.8**	**45.3**	**65.1**

Table 7: Foreground percentage statistics on PointNet++ of different sampling approaches.

SUN RGB-D. We also evaluate our RBGNet against several competing approaches on SUN RGB-D dataset, which is also a standard benchmark for 3D object detection. The results are summarized in Table 3. Our base model with standard PointNet++ achieves 64.1 on mAP@0.25 and 47.2 on mAP@0.5, which outperforms all previous state-of-the-arts point-only methods.

4.4. Ablation Studies and Discussions

In this section, a set of ablative studies are conducted on ScanNet V2 dataset, to investigate effectiveness of essential components of our algorithm. We follows [20] and report the average performance of 25 trials by default.

Effect of ray-based representation. We first ablate the effects of ray-based feature grouping in Table 4, 5 and 6. As evidenced in the first three rows in Table 4, with ray-based feature grouping, our model performs better, *i.e.*, 66.2 → 69.0, 48.2 → 52.9 on mAP@0.25 and mAP@0.5. Note that our model is implemented based on a strong baseline. The first row in Table 4 is actually the VoteNet [28] with corner loss regularization, *vote* sampling in vote aggregation layer

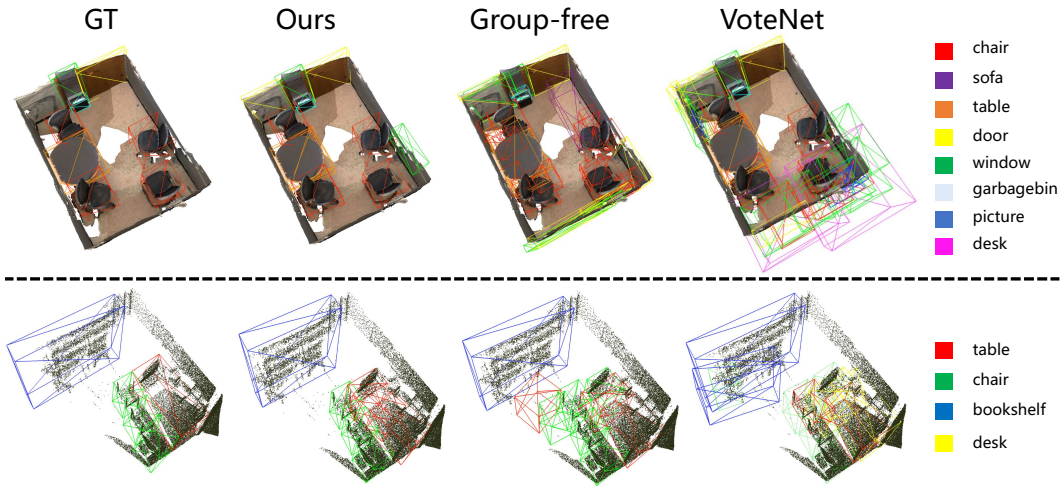

Figure 5: Qualitative results on ScanNet V2 (top) and SUN RGB-D (bottom). The baseline methods are Group-free [20] and VoteNet [28]. Our method can generate high-quality and compact bounding boxes compared with other methods.

method	mAP@0.25	mAP@0.5	frames/s
MLCVNet [42]	64.5	41.4	5.37
BRNet [7]	66.1	50.9	**7.37**
H3DNet [49]	67.2	48.1	3.75
Group-free [20]	67.3	48.9	6.64
Our(R6)	69.0	52.3	7.23
Our(R18)	69.0	52.6	5.70
Our(R38)	69.7	53.3	5.27
Our(R66)	**70.2**	**54.2**	4.75

Table 8: Comparison on realistic inference speed on ScanNet V2. Note that mAP@0.25 and 0.5 are the best results of multiple experiments.

and optimized hyper-parameters. Even on such a strong baseline (almost close to state-of-the-art already, 66.2 vs 66.6 [20] on mAP@0.25), ray-based feature grouping module still boosts our model with a remarkable improvement.

Our ray-based feature grouping module also works well in a wide range of hyper-parameters, such as the number of rays. Table 5 shows its performance with different ray number. More rays can bring significant performance improvement, especially in the mAP@0.5. Compared with the setting without any rays, our ray-102 model performs much better on mAP@0.25 and mAP@0.5 by 2.8 and 4.9, respectively. For the recall of object points, the second column shows that more rays can capture surface points more completely. Considering the trade-off between memory usage and performance improvement, our model finally adopts the variant with 66 rays though more rays is better.

To further demonstrate the effectiveness of ray-based feature grouping module, we refer several grouping strategies in 3D object detection as baselines and compare with them. For a fair comparison, we only switch the feature aggregation mechanism while all other settings remain unchanged. Table 6 shows that our approach achieves more reliable detection results than others with a remarkable margin (1.5 on mAP@0.25 and 3.1 on mAP@0.5).

Effect of foreground biased sampling. Table 4 also demonstrates the effectiveness of the foreground biased sampling strategy. We can observe that, it improves the performance in both settings with and without feature grouping module. This verifies the necessity of sampling more foreground points for 3D object detection tasks. To further ablate the effectiveness of FBS, we compare the foreground points recall of $2^{nd} \rightarrow 4^{th}$ SA layers among different subsampling methods in Table 7. Our sampling strategy draws better performance with a large margin.

4.5. Inference Speed.

The realistic inference speed of our method is competitive with other state-of-the-art methods. For a fair comparison, all experiments are run on the same workstation (single NVIDIA Tesla V100 GPU, 256G RAM, and Xeon E5-2650 v3). The results are shown in Table. 8. Our method achieves better performance with a competitive speed.

5. Conclusion

In this paper, we have presented the RBGNet, a novel framework for 3D object detection from point clouds. We propose the ray-base feature grouping module, which can encode object surface geometry with determined rays and learn better geometric features to boost the performance of point-based 3D detectors. We also introduce the foreground biased sampling to sample more points on object surface while keeping the coverage rate for the whole scene. All of the above designs enable our model to achieve state-of-the-art performance on ScanNet V2 and SUN RGB-D benchmarks with remarkable performance gains.

Acknowledgments. Liwei Wang was supported by National Key R&D Program of China (2018YFB1402600), BJNSF (L172037) and Alibaba Group through Alibaba Innovative Research Program. Project 2020BD006 supported by PKUBaidu Fund.

References

[1] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In *CVPR*, 2018. 3

[2] Ronald T Azuma. A survey of augmented reality. *Presence: teleoperators & virtual environments*, 1997. 1

[3] Mayank Bansal, Alex Krizhevsky, and Abhijit Ogale. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. 2018. 1

[4] Mark Billinghurst, Adrian Clark, and Gun Lee. A survey of augmented reality. 2015. 1

[5] Jintai Chen, Biwen Lei, Qingyu Song, Haochao Ying, Danny Z Chen, and Jian Wu. A hierarchical graph network for 3d object detection on point clouds. In *CVPR*, 2020. 4, 7

[6] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *CVPR*, 2017. 1, 2

[7] Bowen Cheng, Lu Sheng, Shaoshuai Shi, Ming Yang, and Dong Xu. Back-tracing representative points for voting-based 3d object detection in point clouds. In *CVPR*, 2021. 1, 4, 7, 8

[8] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 2, 6, 7

[9] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wen gang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. In *AAAI*, 2021. 1

[10] Oren Dovrat, Itai Lang, and Shai Avidan. Learning to sample. *CoRR*, 2018. 3

[11] Francis Engelmann, Martin Bokeloh, Alireza Fathi, Bastian Leibe, and Matthias Nießner. 3d-mpa: Multi-proposal aggregation for 3d semantic instance segmentation. In *CVPR*, 2020. 7

[12] Benjamin Graham, Martin Engelcke, and Laurens Van Der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *CVPR*, 2018. 2

[13] Ji Hou, Angela Dai, and Matthias Nießner. 3d-sis: 3d semantic instance segmentation of rgb-d scans. In *CVPR*, 2019. 7

[14] Li Jiang, Hengshuang Zhao, Shu Liu, Xiaoyong Shen, Chi-Wing Fu, and Jiaya Jia. Hierarchical point-edge interaction network for point cloud semantic segmentation. In *ICCV*, 2019. 3

[15] Li Jiang, Hengshuang Zhao, Shaoshuai Shi, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Pointgroup: Dual-set point grouping for 3d instance segmentation. In *CVPR*, 2020. 2

[16] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *IROS*, 2018. 1

[17] Itai Lang, Asaf Manor, and Shai Avidan. Samplenet: Differentiable point cloud sampling. In *CVPR*, 2020. 3

[18] Justin Liang, Namdar Homayounfar, Wei-Chiu Ma, Yuwen Xiong, Rui Hu, and Raquel Urtasun. Polytransform: Deep polygon transformer for instance segmentation. In *CVPR*, 2020. 3

[19] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *CVPR*, 2019. 1

[20] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-free 3d object detection via transformers. 2021. 1, 3, 7, 8

[21] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 4, 5

[22] Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3d object detection. In *CVPR*, 2021. 7

[23] Ehsan Nezhadarya, Ehsan Taghavi, Ryan Razani, Bingbing Liu, and Jun Luo. Adaptive hierarchical down-sampling for point cloud classification. In *CVPR*, 2020. 3

[24] Jinhyung Park, Xinshuo Weng, Yunze Man, and Kris Kitani. Multi-modality task cascade for 3d object detection. *arXiv preprint arXiv:2107.04013*, 2021. 7

[25] Sida Peng, Wen Jiang, Huaijin Pi, Xiuli Li, Hujun Bao, and Xiaowei Zhou. Deep snake for real-time instance segmentation. In *CVPR*, 2020. 3

[26] Hughes Perreault, Guillaume-Alexandre Bilodeau, Nicolas Saunier, and Maguelonne Héritier. Centerpoly: real-time instance segmentation using bounding polygons. In *ICCV*, 2021. 3

[27] Charles R Qi, Xinlei Chen, Or Litany, and Leonidas J Guibas. Imvotenet: Boosting 3d object detection in point clouds with image votes. In *CVPR*, 2020. 7

[28] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3d object detection in point clouds. In *ICCV*, 2019. 1, 2, 3, 4, 6, 7, 8

[29] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *CVPR*, 2018. 1, 7

[30] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 1, 2

[31] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 1, 3, 4, 5, 6

[32] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *CVPR*, 2020. 1, 2

[33] Shaoshuai Shi, Li Jiang, Jiajun Deng, Zhe Wang, Chaoxu Guo, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn++: Point-voxel feature set abstraction with local vector representation for 3d object detection. *arXiv preprint arXiv:2102.00463*, 2021. 2

[34] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *CVPR*, 2019. 1, 3, 7

[35] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. *TPAMI*, 2020. 1, 2

[36] Shuran Song, Samuel P. Lichtenberg, and Jianxiong Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *CVPR*, 2015. 2, 6, 7

[37] Shuran Song and Jianxiong Xiao. Deep sliding shapes for amodal 3d object detection in rgb-d images. In *CVPR*, 2016. 1

[38] Dequan Wang, Coline Devin, Qi-Zhi Cai, Philipp

Krähenbühl, and Trevor Darrell. Monocular plan view networks for autonomous driving. In *IROS*, 2019. 1

[39] Haiyang Wang, Wenguan Wang, Xizhou Zhu, Jifeng Dai, and Liwei Wang. Collaborative visual navigation. *arXiv preprint arXiv:2107.01151*, 2021. 1

[40] Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen, and Ping Luo. Polarmask: Single shot instance segmentation with polar representation. In *CVPR*, 2020. 3

[41] Qian Xie, Yu-Kun Lai, Jing Wu, Zhoutao Wang, Dening Lu, Mingqiang Wei, and Jun Wang. Venet: Voting enhancement network for 3d object detection. In *ICCV*, 2021. 7

[42] Qian Xie, Yu-Kun Lai, Jing Wu, Zhoutao Wang, Yiming Zhang, Kai Xu, and Jun Wang. Mlcvnet: Multi-level context votenet for 3d object detection. In *CVPR*, 2020. 3, 4, 7, 8

[43] Wenqiang Xu, Haiyang Wang, Fubo Qi, and Cewu Lu. Explicit shape encoding for real-time instance segmentation. In *ICCV*, 2019. 3

[44] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 2018. 1, 2

[45] Bin Yang, Ming Liang, and Raquel Urtasun. Hdnet: Exploiting hd maps for 3d object detection. In *CoRL*, 2018. 1

[46] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *CVPR*, 2018. 1, 2

[47] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *CVPR*, 2020. 2, 3, 7

[48] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *CVPR*, 2021. 1, 2

[49] Zaiwei Zhang, Bo Sun, Haitao Yang, and Qixing Huang. H3dnet: 3d object detection using hybrid geometric primitives. In *ECCV*, 2020. 1, 3, 7, 8

[50] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. Pointweb: Enhancing local neighborhood features for point cloud processing. In *CVPR*, 2019. 3

[51] Wu Zheng, Weiliang Tang, Sijin Chen, Li Jiang, and Chi-Wing Fu. Cia-ssd: Confident iou-aware single-stage object detector from point cloud. *AAAI*, 2021. 2

[52] Wu Zheng, Weiliang Tang, Li Jiang, and Chi-Wing Fu. Se-ssd: Self-ensembling single-stage object detector from point cloud. In *CVPR*, 2021. 2

[53] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, 2018. 1, 2

[54] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *ICRA*, 2017. 1