AziNorm: Exploiting the Radial Symmetry of Point Cloud for Azimuth-Normalized 3D Perception

Shaoyu Chen [1, 2] Xinggang Wang [1†] Tianheng Cheng [1, 2] Wenqiang Zhang [1]
Qian Zhang [2] Chang Huang [2] Wenyu Liu [1]
[1] School of EIC, Huazhong University of Science & Technology [2] Horizon Robotics
{shaoyuchen,xgwang,thch,wq_zhang,liuwy}@hust.edu.cn {qian01.zhang, chang.huang}@horizon.ai

Abstract

Studying the inherent symmetry of data is of great importance in machine learning. Point cloud, the most important data format for 3D environmental perception, is naturally endowed with strong radial symmetry. In this work, we exploit this radial symmetry via a divide-and-conquer strategy to boost 3D perception performance and ease optimization. We propose Azimuth Normalization (AziNorm), which normalizes the point clouds along the radial direction and eliminates the variability brought by the difference of azimuth. AziNorm can be flexibly incorporated into most LiDAR-based perception methods. To validate its effectiveness and generalization ability, we apply AziNorm in both object detection and semantic segmentation. For detection, we integrate AziNorm into two representative detection methods, the one-stage SECOND detector and the state-of-the-art two-stage PV-RCNN detector. Experiments on Waymo Open Dataset demonstrate that AziNorm improves SECOND and PV-RCNN by 7.03 *mAPH and* 3.01 *mAPH respectively. For segmentation, we integrate AziNorm into KPConv. On SemanticKitti dataset, AziNorm improves KPConv by* 1.6/1.1 *mIoU on val/test set. Besides, AziNorm remarkably improves data efficiency and accelerates convergence, reducing the requirement of data amounts or training epochs by an order of magnitude. SECOND w/ AziNorm can significantly outperform fully trained vanilla SECOND, even trained with only* 10% *data or* 10% *epochs. Code and models are available at* https://github.com/hustvl/AziNorm.

1. Introduction

Environmental perception based on 3D LiDAR point clouds is a fundamental and indispensable ability for autopilot system aiming at high robustness and security. Accurate perception results are the basis of credible motion planning and control, avoiding traffic accidents. And the passed several years have witnessed tremendous performance improvements of LiDAR-based perception. Previous works [24, 25, 29, 30, 42, 44–47, 51] mostly focus on designing network architectures to effectively and efficiently extract semantic information from the point cloud data.

Different from the exhaustively studied network architecture design, the inherent property of LiDAR point cloud has not been explored and exploited yet. LiDAR emits

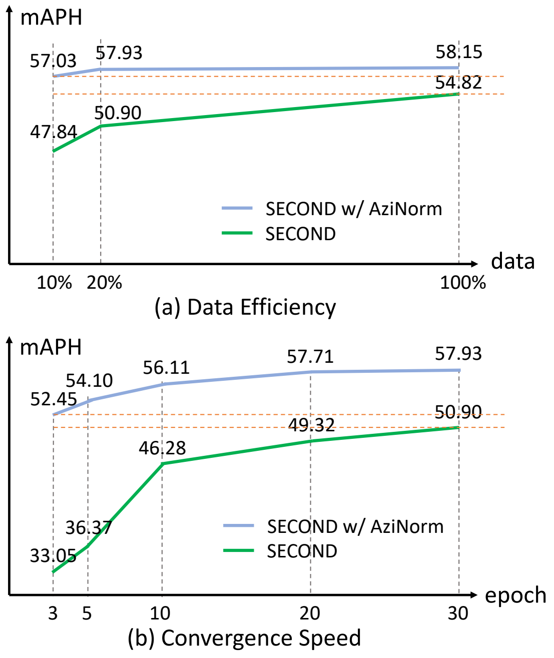

Figure 1. **AziNorm remarkably improves data efficiency (a) and accelerates convergence (b).** With data amounts and training epochs reduced, the performance of SECOND drops dramatically, but SECOND w/ AziNorm still achieves comparable performance. Even trained with only 10% data or 10% epochs, SECOND w/ AziNorm obviously outperforms fully trained vanilla SECOND.

[†]Xinggang Wang is the corresponding author.

laser rays that travel outwards in all azimuths isotropically. Laser rays bounce off surrounding objects, capturing object surface characteristics, and then return to the LiDAR. Both incident rays and reflected rays are along the radial direction. And the generated point clouds are radially symmetrical in a broad sense (Fig. 2).

From the perspective of machine learning, the inherent symmetry of data can serve as a powerful inductive bias for reducing the variability of data and simplifying the recognition system. In this paper, we propose **Azimuth Normalization** (**AziNorm**) to exploit the radial symmetry of point cloud. We split the whole LiDAR scene into individual patches and then normalize the patch's sub point clouds along the radial direction. By normalization, we eliminate the data variability brought by the difference of azimuth and makes perception on normalized point clouds much easier.

AziNorm can be flexibly incorporated into most LiDAR-based perception methods without modifying any implementation detail and significantly boost the performance. To validate its effectiveness and generalization ability, we apply AziNorm in both object detection and semantic segmentation. For detection, we integrate AziNorm into two representative detection methods, the one-stage SECOND [42] detector and the state-of-the-art two-stage PV-RCNN [28] detector. Experiments on large-scale Waymo Open Dataset [33] demonstrate that AziNorm improves SECOND and PV-RCNN by 7.03 mAPH and 3.01 mAPH respectively. For segmentation, we integrate AziNorm into KPConv [36]. On SemanticKitti [1] dataset, AziNorm improves KPConv by $1.6/1.1$ mIoU on val/test set.

More importantly, AziNorm remarkably improves data efficiency and accelerates convergence. Even trained with only 10% data or 10% epochs, SECOND w/ AziNorm significantly outperforms fully trained vanilla SECOND. AziNorm can reduce the requirement of training epochs or data amounts by an order of magnitude, lowering the cost of data acquisition and labeling, which is of great practical value, especially for the data-driven autopilot system.

The main contributions of this paper can be summarized as follows:

- We propose a novel normalization method termed as AziNorm, which exploits the inherent radial symmetry of point clouds to reduce the data variability.
- AziNorm can be easily integrated into most LiDAR-based perception methods and significantly boost the performance. We validate the effectiveness and generalization ability of AziNorm in two perception tasks (object detection and semantic segmentation), based on three methods (SECOND [42], PV-RCNN [28] and KPConv [36]), and on two datasets (Waymo Open Dataset [33] and SemanticKitti [1]).
- AziNorm can reduce the requirement of data amounts or training epochs by an order of magnitude, lowering the cost of data acquisition and labeling, which is of great practical value, especially for the data-driven autopilot system.

In 3D perception, azimuth is the horizontal direction expressed as the angular distance between the direction of observer's heading and the direction of the observed location.

2. Related Work

2.1. Representation Learning on Point Clouds

Representation learning on point clouds has made tremendous progress in recent years [5, 10, 13, 14, 18, 21, 26, 27, 32, 37, 39–41, 50, 51]. The popular PointNet series [26, 27] proposes to directly learn point-wise features from the raw point clouds, where set abstraction operation enables flexible receptive fields by setting different searching radii. 3D sparse convolution [8, 9] is adopted in [30, 42] to effectively learn sparse grid-wise features from the point clouds. [22] combines both grid-based CNN and point-based shared-parameter multi-layer perceptron (MLP) network for point cloud feature learning.

Because of the high complexity and irregularity of point cloud, large amounts of data and long training time are required for learning the representation of point cloud. AziNorm normalizes point clouds and unifies symmetric patterns. It makes representation learning on point clouds much easier and improves both data efficiency and convergence speed.

2.2. 3D Perception on Point Clouds

3D perception on point clouds aims to extract semantic information from the unordered and sparse point cloud data. Existing methods can be divided into three categories, *i.e.*, point-based, grid-based and range-based methods, according to the formats of point cloud they work on.

Grid-based Perception Methods The irregular data format of point cloud brings great challenges for 3D perception. Grid-based methods generally project the point clouds to regular bird-eye view grids [3, 44] or 3D voxels [4, 51] for processing point clouds with 2D/3D CNN. The pioneer work MV3D [3] projects the point clouds to 2D bird-eye view grids and the following works [11, 15, 19, 20, 38, 48] develop better strategies for multi-sensor fusion. [16, 43, 44] introduce more efficient frameworks with bird-eye view representation. Some other works [31, 51] divide the point clouds into 3D voxels to be processed by 3D CNN. PV-RCNN [28] incorporates both 3D voxel CNN and PointNet-based set abstraction for learning discriminative point cloud features. [49, 52] project LiDAR points into polar grids to adapt to the long-tailed distribution of LiDAR points. They help to eliminate the azimuth variability, but their unevenly partitioning the 3D space introduces the problem of scale variance. *I.e.*, in polar grid representation, object's size varies with its distance to LiDAR. Our AziNorm eliminates the azimuth variability without leading to scale variance.

Point-based Perception Methods Point-based perception methods directly operate on the original format of point clouds. F-PointNet [25] first proposes to apply PointNet [26, 27] for 3D detection from the cropped point clouds based on the 2D image bounding boxes. PointRCNN [29] generates 3D proposals directly from point clouds and refines proposals with a RCNN stage. And the following work STD [46] proposes the sparse to dense strategy for better proposal refinement. [24] proposes the hough voting strategy for better object feature grouping. KPConv [36] introduces a novel spatial kernel-based point convolution.

Range-based Perception Methods Range-based perception methods operate on the range image. LaserNet [23] predicts a multimodal distribution for each point to generate the final prediction. RCD [2] learns a dynamic dilation for scale variation and soft range gating for the "boundary blur" issue. RangeDet [6] fixes issues of scale variance and inconsistency between 2D and 3D coordinates. RSN [34] predicts foreground points from range images and applies sparse convolutions on the selected foreground points to detect objects.

AziNorm is with high generalization ability. It can be integrated with grid-based, point-based and some hybrid methods. And existing 3D perception methods treat the LiDAR scene as a whole for processing. AziNorm introduces the divide-and-conquer strategy, which splits the scene into individual patches. With AziNorm, point clouds can be processed in a flexible manner.

2.3. Normalization

Broadly speaking, normalization is to reduce the variability of data. Normalizing the data benefits training a lot [17]. Batch Normalization [12] is a representative normalization technique, which eases optimization and enables deep networks to converge.

In 3D domain, RoI pooling can be considered as an object-level normalization. Some two-stage detectors [28–30] adopt RoI pooling in the second stage, transforming points to the canonical coordinate system, whose XYZ axes are parallel to the predicted 3D bounding boxes. RoI pooling just unifies objects. It does not exploit the property of point cloud and is only applied in object detection. Differently, AziNorm focuses on the inherent radial symmetry of LiDAR point cloud and leverages it for normalization. And AziNorm is a general normalization method, not only for object detection, but also for other LiDAR-based perception tasks. Besides, as other normalization methods do, AziNorm significantly improves optimization.

3. Method

AziNorm functions as a point cloud normalization method to eliminate the variability of azimuth, based on the inherent radial symmetry. We adopt a **divide-and-conquer** strategy to normalize the point clouds along the radial direction. *I.e.*, we split the whole LiDAR scene into patches and normalize in the patch level. We then conduct patch-wise perception and merge patch-wise predictions (see Fig. 2). The details are as follows.

Patch Splitting Firstly, we split the whole LiDAR scene into overlapped circular patches $\{P_i\}$ with stride d and radius r. The layout of patch is optional (discussed in Sec. 4.5) and circular patches are chosen for method description. Each patch corresponds to sub point clouds denoted as $P_i = \{p^i_j, j \in [1, N_i]\}$, where N_i is the number of points in the i-th patch. The center of patch P_i is denoted as c_i. And we denote the azimuth of patch P_i as θ_i, which is the polar angle of the patch center c_i against positive X-axis of the LiDAR coordinate system.

For the whole LiDAR scene, the azimuths vary from $0°$ to $360°$. While for a patch, because of the limited spatial range, the azimuths of all points vary little and the difference is negligible. Thus, all the points in a patch can be treated as a whole for normalization.

Patch Selection In the scenario of autopilot, the LiDAR scene is of large range and high sparsity (see Fig. 4). Taking Waymo [33] dataset for example, in its extremely large scene of $150m \times 150m$, about 80% of areas are void or only contain few ground points.

Existing 3D perception methods have to regard the complete point clouds as a whole for processing. With AziNorm, we can process point clouds in a lower patch level and in a flexible manner. And the sparsity of LiDAR scene can be leveraged to achieve higher efficiency.

We adopt two patch selection strategies, **negative filtering** and **positive sampling**, for more efficient training and inference. Specifically, in both training and inference, we directly filter out patches only with negligible points, avoiding time consumption in background regions. Besides, for object detection, in the training phase, we keep all foreground patches that contain ground truth objects and only sample a few background patches, balancing the ratio between positive and negative samples.

Patch Normalization After patch splitting and selection, patch-specific transformation is conducted to normalize among different patches.

For every patch, we build a patch coordinate system, whose positive X-axis is along the radial direction (see Fig. 2). Within patch i, we transform points from the LiDAR coordinate system to the patch coordinate system with the extrinsic matrix R_{θ_i} and T_{c_i},

$$\bar{p}^i_j = R_{\theta_i} \cdot (p^i_j + T_{c_i}). \tag{1}$$

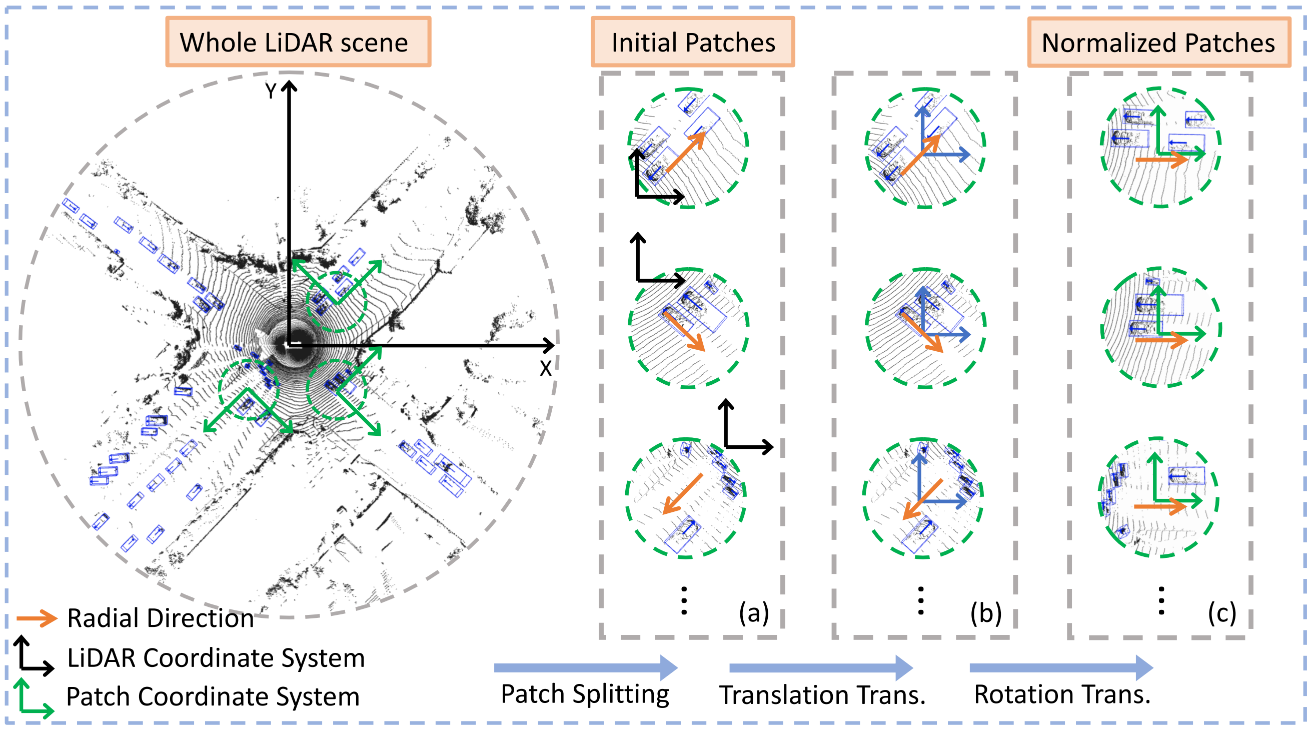

Figure 2. **Illustration of Azimuth Normalization.** The whole LiDAR scene is split into overlapped circular patches. Within each patch, we transform the coordinates of the sub point clouds from the LiDAR coordinate system to the specific patch coordinate system (translation and rotation transformation sequentially). With AziNorm, the radial directions of all patches are unified and the variability of azimuth is eliminated. Perception based on normalized sub point clouds is significantly simplified. For clarity, only three typical patches are visualized. In (a), (b), and (c), the coordinates of point clouds are relative to the LiDAR coordinate system, the temporary coordinate system colored in blue, and the patch coordinate system, respectively.

R_{θ_i} and T_{c_i} are the rotation matrix determined by θ_i and translation matrix determined by c_i respectively. $\bar{p}^i_j$ is the transformed coordinate of p^i_j. $\bar{p}^i_j$ and p^i_j are relative to the patch coordinate system and LiDAR coordinate system respectively. After patch-specific transformation, the radial directions of all patches are unified (see Fig. 2(c)). The variability of azimuth is almost eliminated and the point clouds are simplified.

Patch-wise Perception The next step is applying perception algorithms on normalized patches. Patch's sub point clouds keep the same data format and geometric information with the original point clouds. Thus, perception on patches is the same with perception on the whole LiDAR scene. Most off-the-shelf LiDAR-based perception methods, *e.g.*, both object detection and semantic segmentation, both point-based [24, 25, 29, 45, 46] and grid-based [30, 42, 44, 47, 51] ones, are applicable for patch-wise perception.

Specifically, we treat every patch as independent point clouds and concatenate all patches in the batch dimension. Then a chosen perception method is adopted for both training and inference. It's worth noting that we need not make any special modification for adapting to different perception methods. All the hyper-parameters (including voxel size, batch size, learning rate, *etc.*) are kept the same with the off-the-shelf configurations.

Inverse Normalization For object detection, with the detection on patches finished, we get detected 3D bounding boxes, represented in different patch coordinate systems. We then conduct inverse normalization to convert the patch detection results $\{\bar{B}_i\}$ back to the original LiDAR coordinate system representation $\{B_i\}$, which is formulated as,

$$B_i = R^{-1}_{\theta_i} \cdot \bar{B}_i - T_{c_i}. \tag{2}$$

For semantic segmentation, after applying segmentation methods on patches, we get predicted labels of every point of every patch. We then map the predicted results from patches to the original point clouds.

Patch Merging Patches are overlapped and the predictions are duplicated. We merge patch-wise predictions to generate the final predictions.

For object detection, through strict non-maximum suppression (NMS), duplicated bounding box predictions are filtered out. For semantic segmentation, one point is included by several patches. For each point, we just average the predictions of different patches.

4. Experiments

4.1. AziNorm-based Object Detection

Experimental Settings AziNorm can be flexibly incorporated into most point cloud based object detection methods. For validating its effectiveness and generalization ability, we integrate AziNorm into two representative detection methods. One is the one-stage detector SECOND [42] with a compact and efficient pipeline, and the other is the two-stage PV-RCNN [28], which is the state-of-the-art 3D detection method. For fair comparison, we keep all the hyper-parameters the same with the off-the-shelf configurations provided by OpenPCDet [35] and do not modify any implementation detail of the detector. As for patch splitting, unless otherwise specified, the patch's radius r is $9.6m$ and the stride d between two adjacent patches is $6.4m$. Experiments are conducted with 8 RTX 3090 GPUs.

Dataset and Metrics We conduct experiments on Waymo Open Dataset [33], which is the largest public dataset for autonomous driving to date. There are totally 798 training sequences with around $160k$ LiDAR frames, and 202 validation sequences with $40k$ LiDAR frames. Using the entire training set requires a lot of computing resources and training time. Following [28], to efficiently conduct experiments, we uniformly sample 20% frames (about $32k$ frames) for training (except for the experiments of data efficiency). We report mean average precision (mAP) and mean average precision weighted by heading (mAPH) on two difficulty levels (LEVEL 1 and LEVEL 2). The 3D IoU threshold is set as 0.7 for vehicle detection and 0.5 for pedestrian/cyclist detection.

SECOND w/ AziNorm As shown in Tab. 1, compared with vanilla SECOND, SECOND w/ AziNorm achieves remarkably better mAP/mAPH on all difficulty levels for the detection of all three categories. And AziNorm improves the recall under the IoU threshold of 0.3, 0.5 and 0.7 by 1.37, 2.57 and 3.07 respectively (Tab. 2). Notably, the performance of pedestrian and cyclist is improved more. We can observe a gain of 9.23 and 8.41 APH of LEVEL 2 difficulty for pedestrian and cyclist respectively. Pedestrian and cyclist occupy less space in real world and are with fewer LiDAR points on their surfaces. In 3D detection, these two classes are more challenging. AziNorm unifies the symmetric point cloud patterns and highly reduces the learning difficulty of hard cases with few LiDAR points. Thus, the detection of pedestrian and cyclist benefits more from AziNorm.

PV-RCNN w/ AziNorm As shown Tab. 1, when incorporated into the state-of-the-art detector PV-RCNN with less room for improvement, AziNorm still brings significant gain. PV-RCNN w/ AziNorm outperforms vanilla PV-RCNN in all metrics. As for the mAPH of LEVEL 2 difficulty of all classes, AziNorm improves PV-RCNN by 3.01 mAPH. And AziNorm improves the recall under the IoU threshold of 0.3, 0.5 and 0.7 by 4.04, 4.41 and 4.69 respectively (Tab. 2).

4.2. AziNorm-based Semantic Segmentation

Experimental Settings For further validating the effectiveness and generalization ability, we integrate AziNorm into a widely-used semantic segmentation method named KPConv [36]. We follow the released official code and only ablate on AziNorm. All hyper-parameters are kept the same for fair comparison.

Dataset and Metrics We conduct experiments on SemanticKitti [1], which is a large-scale dataset based on the KITTI Vision Benchmark [7] with dense point-wise annotations for the complete 360° FOV. As official guidance [1] suggests, we use mean intersection-over-union (mIoU) over all classes as the evaluation metric.

KPConv w/ AziNorm Tab. 3 and Tab. 4 are the results of SemanticKitti. AziNorm-based KPConv significantly outperforms vanilla KPConv by 1.6% and 1.1% mIoU on val and test sets respectively. The experiments prove that AziNorm works in both object detection and semantic segmentation tasks. AziNorm is highly compatible and can also be applied in other LiDAR-based perception tasks.

4.3. Data Efficiency & Convergence

Apart from boosting perception performance, AziNorm can also significantly improve data efficiency and accelerates convergence. In Fig. 1, under different data amounts and training epochs, we compare the performance (mAPH of LEVEL 2 difficulty for all classes) of SECOND detector w/ AziNorm and w/o AziNorm on Waymo. All other hyper-parameters are kept the same. With data amounts and training epochs reduced, the performance of SECOND drops dramatically, but SECOND w/ AziNorm still achieves comparable performance. SECOND w/ AziNorm trained with only 3 epochs outperforms vanilla SECOND trained with 30 epochs. And SECOND w/ AziNorm trained with only 10% data outperforms vanilla SECOND trained with 100% data. AziNorm significantly improves data efficiency and accelerates convergence, reducing the requirement of both data amounts and training epochs by an order of magnitude.

From the perspective of machine learning, a perception method aims at approximating a function that maps from the point cloud patterns to the labels (*e.g.*, 3D bounding boxes for detection and point-wise class labels for segmentation). AziNorm treats the radial symmetry as an important inductive bias of the system. It unifies symmetric patterns together and reduces the variability of input data. With less variability in input, the mapping function is highly simpli-

Difficulty	Method	All		Vehicle		Pedestrian		Cyclist	
		mAP	mAPH	AP	APH	AP	APH	AP	APH
LEVEL 1	SECOND [42]	60.57	56.49	68.15	67.56	58.20	47.94	55.35	53.98
	SECOND w/ AziNorm	67.32	63.40	70.73	70.24	67.39	57.23	63.84	62.74
	Improvement	+6.75	+6.91	+2.58	+2.68	+9.19	+9.29	+8.49	+8.76
	PV-RCNN [28]	69.74	65.23	74.27	73.58	70.39	59.10	64.57	63.00
	PV-RCNN w/ AziNorm	72.64	68.32	75.17	74.64	75.05	63.92	67.69	66.41
	Improvement	+2.90	+3.09	+0.90	+1.06	+4.66	+4.82	+3.12	+3.41
LEVEL 2	SECOND [42]	54.48	50.90	59.61	59.09	50.22	41.32	53.62	52.28
	SECOND w/ AziNorm	61.51	57.93	63.03	62.56	59.73	50.55	61.76	60.69
	Improvement	+7.03	+7.03	+3.42	+3.47	+9.51	+9.23	+8.14	+8.41
	PV-RCNN [28]	63.10	59.04	65.30	64.68	61.33	51.31	62.67	61.14
	PV-RCNN w/ AziNorm	65.95	62.05	66.27	65.80	65.93	55.97	65.65	64.39
	Improvement	+2.85	+3.01	+0.97	+1.12	+4.60	+4.66	+2.98	+3.25

Table 1. **Performance comparisons of object detection on the Waymo [33] val set.** All experiments are conducted with the same configuration based on the official codebase OpenPCDet [35]. AziNorm-based detectors significantly outperform their baselines in all metrics of all difficulty levels.

Method	Rec.@0.3	Rec.@0.5	Rec.@0.7
SECOND [42]	80.13	70.37	44.60
SECOND w/ AziNorm	81.50	72.94	47.67
Improvement	+1.37	+2.57	+3.07
PV-RCNN [28]	84.86	78.12	53.92
PV-RCNN w/ AziNorm	88.90	82.53	58.61
Improvement	+4.04	+4.41	+4.69

Table 2. **Recall comparisons on the Waymo [33] val set under IoU thresholds of 0.3, 0.5 and 0.7.** AziNorm-based detectors achieve much higher recall rates than their baselines under all IoU thresholds.

fied and easier to be approximated. Consequently, both the data efficiency and convergence speed are significantly improved.

4.4. Limitation

Overhead of AziNorm AziNorm brings some computation overhead because of the overlap among patches. But it is actually a problem of speed-accuracy trade-off. We can flexibly adjust the degree of overlap to balance between performance and inference speed.

For offline applications, *e.g.*, constructing the high-definition map or generating 3D labels for camera-based systems (3D Data Auto Labeling), perception performance has the highest priority and latency is not restricted. We can make patches highly overlap to reach the performance upper bound of AziNorm.

For real-time applications, *e.g.*, real-time perception in autopilot system, it's feasible to reduce the overlap of patch for faster inference. AziNorm can still bring gain even with little overlap. In Tab. 5, we provide the speed comparison between SECOND and SECOND w/ AziNorm under the setting of little overlap ($r = 11.2m$, $d = 18.8m$, trained for 5 epochs). SECOND w/ AziNorm achieves better performance and comparable inference speed.

4.5. Ablation Study

In this section, we ablate on detailed elements of AziNorm on Waymo Open Dataset [33], to validate the design of AziNorm. SECOND [42] detector is chosen for the experiments, given its compact and efficient pipeline.

Patch Splitting, Translation Trans. and Rotation Trans. In Tab. 6, we provide ablation studies about patch splitting, translation transformation and rotation transformation, to show what makes AziNorm work. The experiments are trained for 5 epochs. The patch splitting mechanism can be considered as a kind of sampling strategy or data augmentation, and translation transformation narrows the numeric range of point coordinate and benefits convergence. Thus, patch splitting and translation transformation both bring gain. But their gain is limited, and the improvement of AziNorm mainly comes from rotation transformation. Rotation transformation is the key step of AziNorm, which normalizes the patches along the radial direction. It brings the most gain as expected, validating the motivation of this work, *i.e.*, exploiting the radial symmetry of point cloud.

Radius r and Stride d Ablation studies about radius r and stride d are reported in Tab. 7 and Tab. 8 (trained for 5 epochs). The performance of AziNorm is robust to radius r and stride d in a wide range. And when applied to different datasets, there is no need to specially tune r and d.

Patch Layout Ablation studies about the layout of patch are reported in Tab. 9 (trained for 5 epochs). For fair com-

Method	mIoU	car	bcycle.	mcycle.	truck	o-veh.	person	bclist.	mclist.	road	parking	side.	o-gro.	bui.	fence	vege.	trunk	terrain	pole	tra.
KPConv	61.3	95.5	39.1	62.6	61.0	50.7	72.1	91.5	0.00	91.1	30.2	80.3	1.1	89.8	70.1	88.5	68.7	74.0	65.7	32.9
KPConv w/ AziNorm	62.9	95.9	42.6	66.3	67.0	55.1	73.0	90.3	0.00	91.4	33.3	80.5	2.7	90.1	70.7	88.9	70.7	74.6	66.7	36.1
Improvement	+1.6	+0.4	+3.5	+3.7	+6.0	+4.4	+0.9	-1.2	+0.0	+0.3	+3.1	+0.2	+1.6	+0.3	+0.6	+0.4	+2.0	+0.6	+1.0	+3.2

Table 3. **Performance comparisons of semantic segmentation on the SemanticKitti [1] val set**. All experiments are conducted with the same configuration based on the official code of KPConv [36]. AziNorm-based KPConv significantly outperforms vanilla KPConv.

Method	mIoU	car	bcycle.	mcycle.	truck	o-veh.	person	bclist.	mclist.	road	parking	side.	o-gro.	bui.	fence	vege.	trunk	terrain	pole	tra.
KPConv	52.9	87.8	43.2	46.5	32.6	40.1	56.4	59.9	10.1	80.0	49.2	67.9	25.5	78.3	59.4	71.8	54.7	58.9	48.7	34.4
KPConv w/ AziNorm	54.0	88.5	41.2	46.8	37.0	42.5	57.9	59.7	11.0	80.8	52.8	68.5	27.8	79.4	59.5	72.7	55.7	59.6	48.7	35.6
Improvement	+1.1	+0.7	-2.0	+0.3	+0.8	+2.4	+1.5	-0.2	+0.9	+0.8	+3.6	+0.6	+2.3	+1.1	+0.1	+0.9	+1.0	+0.7	+0.0	+1.1

Table 4. **Performance comparisons of semantic segmentation on the SemanticKitti [1] test set.** All experiments are conducted with the same configuration based on the official code of KPConv [36]. AziNorm-based KPConv significantly outperforms vanilla KPConv.

Method	L1 mAP	L1 mAPH	L2 mAP	L mAPH	FPS
SECOND	51.99	45.16	46.35	40.31	27
SECOND w/ AziNorm	56.55	51.10	50.65	45.89	24

Table 5. **Inference speed comparison**. Tested on RTX 3090. Batch size is 1. SECOND w/ AziNorm achieves better performance and comparable inference speed.

Method	L1 mAP	L1 mAPH	L2 mAP	L2 mAPH
SECOND	51.99	45.16	46.35	40.31
+ Patch Splitting	53.07	46.87	47.52	42.04
+ Translation Trans.	54.18	47.83	47.67	43.10
+ Rotation Trans.	64.74	59.99	58.41	54.22

Table 6. **Ablation studies about patch splitting, translation transformation and rotation transformation.** The improvement of AziNorm mainly comes from rotation transformation, which normalizes the point clouds along the radial direction.

Method	r	L1 mAP	L1 mAPH	L2 mAP	L2 mAPH
SECOND	-	51.99	45.16	46.35	40.31
SECOND w/ AziNorm	8.0m	63.44	58.43	57.17	52.77
SECOND w/ AziNorm	9.6m	64.74	59.99	58.41	54.22
SECOND w/ AziNorm	11.2m	65.54	60.34	59.13	54.54

Table 7. **Ablation studies about radius r.** Stride d is set as $6.4m$. The performance of AziNorm is robust to radius r in a wide range.

Method	d	L1 mAP	L1 mAPH	L2 mAP	L2 mAPH
SECOND	-	51.99	45.16	46.35	40.31
SECOND w/ AziNorm	8.0m	63.56	58.71	57.53	53.15
SECOND w/ AziNorm	7.2m	64.74	59.64	58.44	53.95
SECOND w/ AziNorm	6.4m	64.74	59.99	58.41	54.22
SECOND w/ AziNorm	5.6m	65.41	60.16	59.06	54.44

Table 8. **Ablation studies about radius d.** Radius r is set as $9.6m$. The performance of AziNorm is robust to stride d in a wide range.

parison, we make the areas of these two kinds of patch sim-

Patch layout	Area	L1 mAP	L1 mAPH	L2 mAP	L2 mAPH
Circular	290m^2 (r=9.6m)	64.74	59.99	58.41	54.22
Square	310m^2 (a=17.6m)	65.41	60.45	59.04	54.66

Table 9. **Ablation studies about the layout of patch.** Stride d is $6.4m$. "a" denotes the length of side. AziNorm is robust to the choice of patch layout.

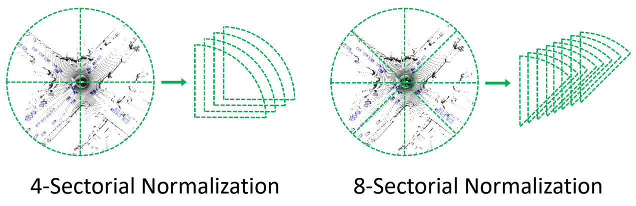

Figure 3. **4- and 8-Sectorial Normalization.** The LiDAR scene is split into sectorial regions for normalization.

ilar. Circular patches and square patches have similar performance. Thus, AziNorm is robust to the choice of patch layout. We adopt circular patches for method description.

Sectorial Normalization Apart from AziNorm, we also design Sectorial Normalization (see Fig. 3). The LiDAR scene is split into sectorial regions for normalization. Adjacent sectorial regions are slightly overlapped. Tab. 10 shows the comparisons between Sectorial Normalization and AziNorm (based on SECOND and trained for 30 epochs). SECOND w/ 4-Sectorial Normalization outperforms vanilla SECOND w/o normalization by 1.47 mAPH of LEVEL 2 difficulty for all classes. And SECOND w/ 8-Sectorial Normalization achieves a higher improvement of 1.91 mAPH. AziNorm achieves the highest improvement of 7.03 mAPH.

Compared with the baseline (w/o norm.), Sectorial Normalization changes little, no sampling strategy, no translation transformation and negligible computation overhead. It just splits the scene into sectorial regions and rotates them.

Difficulty	Method	All		Vehicle		Pedestrian		Cyclist	
		mAP	mAPH	AP	APH	AP	APH	AP	APH
LEVEL 1	w/o Norm.	60.57	56.49	68.15	67.56	58.20	47.94	55.35	53.98
	4-Sectorial Norm.	61.58	57.91	69.26	68.70	58.91	49.57	56.58	55.45
	8-Sectorial Norm.	62.21	58.36	68.78	68.21	60.32	50.71	57.53	56.15
	AziNorm	**67.32**	**63.40**	**70.73**	**70.24**	**67.39**	**57.23**	**63.84**	**62.74**
LEVEL 2	w/o Norm.	54.48	50.90	59.61	59.09	50.22	41.32	53.62	52.28
	4-Sectorial Norm.	55.66	52.37	60.66	60.16	51.50	43.24	54.82	53.72
	8-Sectorial Norm.	56.26	52.81	60.18	59.68	52.84	44.32	55.77	54.43
	AziNorm	**61.51**	**57.93**	**63.03**	**62.56**	**59.73**	**50.55**	**61.76**	**60.69**

Table 10. **Experiments of Sectorial Normalization and AziNorm based on SECOND [42] on the Waymo [33] val set.** The comparisons between Sectorial Normalization and baseline validates the potential of the radial symmetry. And comparisons between AziNorm and 4-/8-Sectorial Normalization prove that finer normalization granularity brings more significant gain.

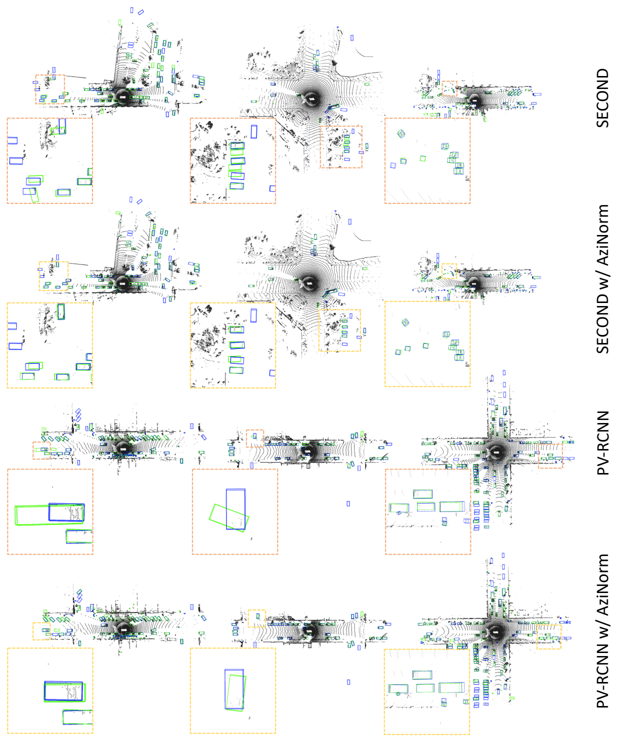

Figure 4. **Qualitative comparisons of AziNorm based on SECOND [42] and PV-RCNN [28].** Blue and green boxes are ground-truth annotations and predicted results respectively. When integrated with AziNorm, SECOND and PV-RCNN both achieve more accurate bounding box predictions, especially for the direction of bounding box. Key regions are zoomed in for better view.

With this simple change, we can get considerable gain. It proves the idea of exploiting the radial symmetry is effective and potential.

Besides, the experiments prove that finer **normalization granularity** brings more significant gain. 4- and 8-Sectorial Normalization reduce the azimuth variability to a range of 90° and 45° respectively. They help to simplify the point clouds but the normalization granularity is limited. AziNorm corresponds to a finer normalization granularity, *i.e.*, patch. The azimuth variability within a patch is negligible because of the limited spatial range of patch. And the variability among patches is totally normalized through patch-specific transformation. Thus, AziNorm almost eliminates the azimuth variability of the whole scene. It makes the most of the radial symmetry and achieves the most significant improvements.

4.6. Qualitative Comparisons

In Fig 4, we provide qualitative comparisons to further demonstrate the effectiveness of AziNorm. When integrated with AziNorm, SECOND and PV-RCNN both achieve more accurate bounding box predictions, especially for the direction of bounding box, which is quite important for predicting the driver's intention in autopilot.

5. Conclusion

We present AziNorm, which exploits the inherent radial symmetry to normalize point clouds along the radial direction and eliminate the variability brought by the difference of azimuth. AziNorm is of great practical value, especially for autopilot and robotics. AziNorm 1) significantly boosts the perception performance, 2) improves data efficiency and lowers the cost of data acquisition and labeling, 3) accelerates convergence and saves training time. Moreover, AziNorm is highly extensible. It can be easily combined with many perception tasks (detection, segmentation, etc.), and sensors (LiDAR, RADAR, surround-view cameras, etc.) that possess the same radial symmetry.

Acknowledgement This work was in part supported by NSFC (No. 61733007 and No. 61876212) and the Zhejiang Laboratory under Grant 2019NB0AB02.

References

[1] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jürgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *ICCV*, 2019. 2, 5, 7

[2] Alex Bewley, Pei Sun, Thomas Mensink, Dragomir Anguelov, and Cristian Sminchisescu. Range conditioned dilated convolutions for scale invariant 3d object detection. *arXiv:2005.09927*, 2020. 3

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

[4] Yilun Chen, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Fast point r-cnn. In *ICCV*, 2019. 2

[5] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, 2019. 2

[6] Lue Fan, Xuan Xiong, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Rangedet: In defense of range view for lidar-based 3d object detection. In *ICCV*, 2021. 3

[7] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *IJRR*, 2013. 5

[8] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. *CVPR*, 2018. 2

[9] Benjamin Graham and Laurens van der Maaten. Submanifold sparse convolutional networks. *arXiv:1706.01307*, 2017. 2

[10] Qiangui Huang, Weiyue Wang, and Ulrich Neumann. Recurrent slice networks for 3d segmentation of point clouds. In *CVPR*, 2018. 2

[11] Tengteng Huang, Zhe Liu, Xiwu Chen, and Xiang Bai. Epnet: Enhancing point features with image semantics for 3d object detection. In *ECCV*, 2020. 2

[12] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 3

[13] Maximilian Jaritz, Jiayuan Gu, and Hao Su. Multi-view pointnet for 3d scene understanding. In *ICCVW*, 2019. 2

[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. 2

[15] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven Waslander. Joint 3d proposal generation and object detection from view aggregation. *IROS*, 2018. 2

[16] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. *CVPR*, 2019. 2

[17] Yann LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural Networks: Tricks of the Trade - Second Edition*. 2012. 3

[18] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *NeurIPS*, 2018. 2

[19] Ming Liang*, Bin Yang*, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *CVPR*, 2019. 2

[20] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *ECCV*, 2018. 2

[21] Ze Liu, Han Hu, Yue Cao, Zheng Zhang, and Xin Tong. A closer look at local aggregation operators in point cloud analysis. *arXiv:2007.01294*, 2020. 2

[22] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. In *NeurIPS*, 2019. 2

[23] Gregory P. Meyer, Ankit Laddha, Eric Kee, Carlos Vallespi-Gonzalez, and Carl K. Wellington. Lasernet: An efficient probabilistic 3d object detector for autonomous driving. In *CVPR*, 2019. 3

[24] 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, 3, 4

[25] 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, 3, 4

[26] 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. 2, 3

[27] 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. 2, 3

[28] 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. 2, 3, 5, 6, 8

[29] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *CVPR*, 2019. 1, 3, 4

[30] 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, 3, 4

[31] Shuran Song and Jianxiong Xiao. Deep sliding shapes for amodal 3d object detection in rgb-d images. In *CVPR*, 2016. 2

[32] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *CVPR*, 2018. 2

[33] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, 2020. 2, 3, 5, 6, 8

[34] Pei Sun, Weiyue Wang, Yuning Chai, Gamaleldin Elsayed, Alex Bewley, Xiao Zhang, Cristian Sminchisescu, and Dragomir Anguelov. RSN: range sparse net for efficient, accurate lidar 3d object detection. In *CVPR*, 2021. 3

[35] OpenPCDet Development Team. Openpcdet: An open-source toolbox for 3d object detection from point clouds. https://github.com/open-mmlab/OpenPCDet, 2020. 5, 6

[36] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, 2019. 2, 3, 5, 7

[37] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, Francois Goulette, and Leonidas J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, 2019. 2

[38] Sourabh Vora, Alex H Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential fusion for 3d object detection. In *CVPR*, 2020. 2

[39] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *TOG*, 2019. 2

[40] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *CVPR*, 2019. 2

[41] Qiangeng Xu, Xudong Sun, Cho-Ying Wu, Panqu Wang, and Ulrich Neumann. Grid-gcn for fast and scalable point cloud learning. In *CVPR*, 2020. 2

[42] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 2018. 1, 2, 4, 5, 6, 8

[43] Bin Yang, Ming Liang, and Raquel Urtasun. Hdnet: Exploiting hd maps for 3d object detection. In *2nd Conference on Robot Learning*, 2018. 2

[44] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *CVPR*, 2018. 1, 2, 4

[45] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *CVPR*, 2020. 1, 4

[46] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. STD: sparse-to-dense 3d object detector for point cloud. *ICCV*, 2019. 1, 3, 4

[47] Maosheng Ye, Shuangjie Xu, and Tongyi Cao. Hvnet: Hybrid voxel network for lidar based 3d object detection. In *CVPR*, 2020. 1, 4

[48] Jin Hyeok Yoo, Yeocheol Kim, Ji Song Kim, and Jun Won Choi. 3d-cvf: Generating joint camera and lidar features using cross-view spatial feature fusion for 3d object detection. In *ECCV*, 2020. 2

[49] Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, Boqing Gong, and Hassan Foroosh. Polarnet: An improved grid representation for online lidar point clouds semantic segmentation. In *CVPR*, 2020. 2

[50] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. Pointweb: Enhancing local neighborhood features for point cloud processing. In *CVPR*, 2019. 2

[51] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, 2018. 1, 2, 4

[52] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. *arXiv:2011.10033*, 2020. 2