Supplementary Material - Investigating the Impact of Multi-LiDAR Placement on Object Detection for Autonomous Driving

Hanjiang Hu[1*] Zuxin Liu[1*] Sharad Chitlangia[2†] Akhil Agnihotri[3†] Ding Zhao[1]
[1]Carnegie Mellon University [2]Amazon [3]University of Southern California
{hanjianghu,zuxinl,dingzhao}@cmu.edu, chitshar@amazon.com, akhil.agnihotri@usc.edu

1. Details of Experimental Settings

In the supplementary material, we first add more details about the setting in our experiments.

1.1. Data Collection Details in CARLA

We choose four different Towns for data collection from CARLA v0.9.10, which are shown in Figure 1 and the number of frames in each town is about 11000 with 8 different routes covering all the main roads. To split objects, since CARLA itself does not separate Car, Van and Cyclist from Vehicles, we spawn all types of Vehicles and manually denote Van and Cyclist with the actor IDs of *carlamotors.carlacola*, *harley-davidson.low_rider*, *diamondback.century*, *yamaha.yzf*, *bh.crossbike*, *kawasaki.ninja* and *gazelle.omafiets* while denoting the remaining as Cars. Note the box truck for *carlamotors.carlacola* is with size over $5.2m \times 2.4m \times 2.6m$, which is the only too large van and categorized into Van and Cyclist, occupying about one tenth of frames in the abnormal-size class.

When collecting point cloud, we choose the frequency of simulation frame to be 20Hz for synchronization and run CARLA on two NVIDIA GeForce RTX 3090 GPUs with RAM 120G in a Ubuntu 18.04 docker container.

1.2. LiDAR Placement Details

The ego-vehicle has its coordinate frame at its geometric center at $[40, 20, 0, 0, 0]$ with respect to the ROI frame of reference. All LiDAR configurations are illustrated in Figure 3, and their detailed coordinates are given in Table 1. These coordinates are with respect to the ego vehicle's coordinate frame and will be transformed to ROI framework, as shown in Figure 2. Note that the coordinate frames in CARLA are left-handed, and we make the extra transformation in ROI when calculating the POG and surrogate metric.

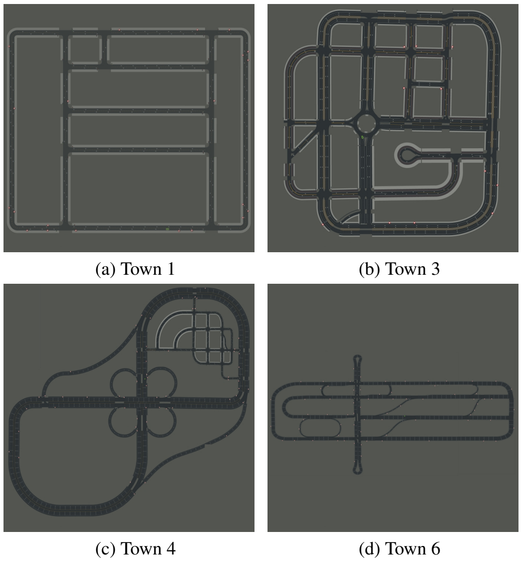

(a) Town 1
(b) Town 3
(c) Town 4
(d) Town 6

Figure 1. The four diverse town maps we used to collect data and conduct experiments in CARLA v0.9.10.

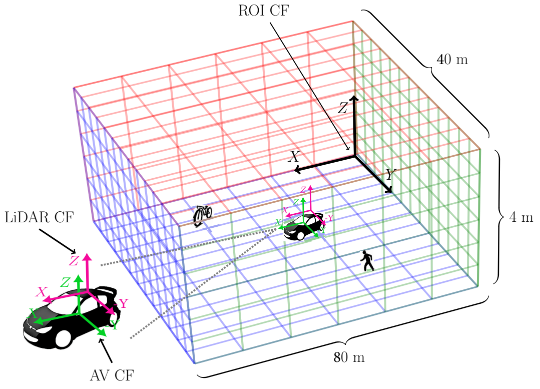
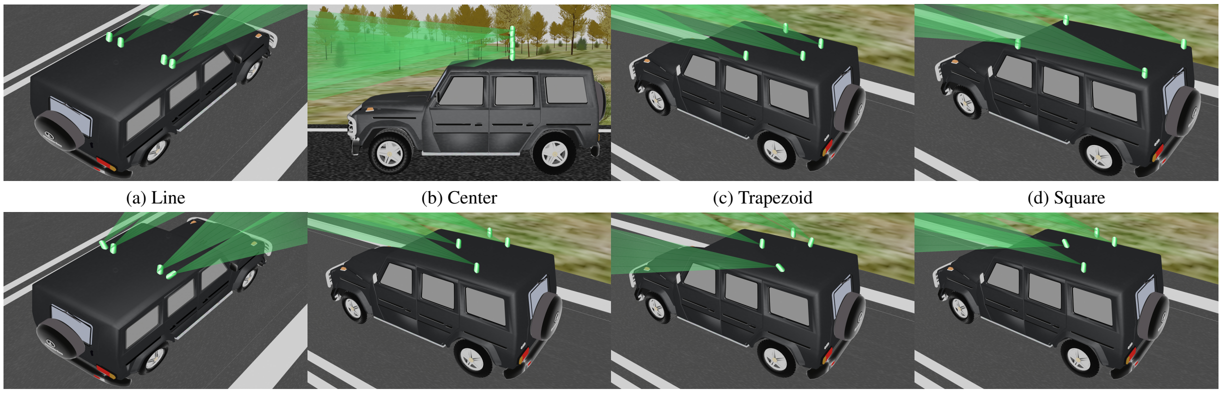


Figure 2. Coordinate frames of ROI, the ego-vehicle, and the LiDAR. Figure not to scale.

1.3. Training Details of Detection Algorithms

We use representative algorithms from OpenPCDet [4] to evaluate the performance under different LiDAR place-

[*] equal contribution
[†] work done while interning at CMU, prior to current affiliations



(a) Line (b) Center (c) Trapezoid (d) Square

(e) Line-roll (f) Pyramid (g) Pyramid-roll (h) Pyramid-pitch

Figure 3. Illustrations of different multi-LiDAR placements used in the experiments

Placement	x	y	z	roll	pitch
Line	-0.0	-0.6	2.2	0.0	0.0
	0.0	-0.4	2.2	0.0	0.0
	0.0	0.4	2.2	0.0	0.0
	0.0	0.6	2.2	0.0	0.0
Center	0.0	0.0	2.4	0.0	0.0
	0.0	0.0	2.6	0.0	0.0
	0.0	0.0	2.8	0.0	0.0
	0.0	0.0	3.0	0.0	0.0
Trapezoid	-0.4	0.2	2.2	0.0	0.0
	-0.4	-0.2	2.2	0.0	0.0
	0.2	0.5	2.2	0.0	0.0
	0.2	-0.5	2.2	0.0	0.0
Square	-0.5	0.5	2.2	0.0	0.0
	-0.5	- 0.5	2.2	0.0	0.0
	0.5	0.5	2.2	0.0	0.0
	0.5	-0.5	2.2	0.0	0.0
Line-roll	- 0.0	-0.6	2.2	-0.28	0.0
	0.0	-0.4	2.2	0.0	0.0
	0.0	0.4	2.2	0.0	0.0
	0.0	0.6	2.2	0.28	0.0
Pyramid	-0.2	-0.6	2.2	0.0	0.0
	0.4	0.0	2.4	0.0	0.0
	-0.2	0.0	2.6	0.0	0.0
	-0.2	0.6	2.2	0.0	0.0
Pyramid-roll	-0.2	-0.6	2.2	-0.28	0.0
	0.4	0.0	2.4	0.0	0.0
	-0.2	0.0	2.6	0.0	0.0
	-0.2	0.6	2.2	0.28	0.0
Pyramid-pitch	-0.2	-0.6	2.2	0.0	0.0
	0.4	0.0	2.4	0.0	-0.09
	-0.2	0.0	2.6	0.0	0.0
	-0.2	0.6	2.2	0.0	0.0

Table 1. Coordinates of LiDAR sensors with respect to the ego-vehicle coordinate frame. All values of *x,y,z* are in meters and roll and pitch angles are in *rad*.

Hyperparameter	Value
Epochs	10
Optimizer	adam_onecycle
Learning Rate	0.01
Weight Decay:	0.01
Momentum:	0.9
Learning Rate Clip	0.0000001
Learning Rate Decay	0.1
Div Factor	10
Warmup Epoch	1
Learning Rate Warmup	False
Gradient Norm Clip	10
MOMS	[0.95, 0.85]
PCT_START	0.1

Table 2. Hypeparameters for optimization in model training

ments. We keep all the models' hyperparameters the same as the default KITTI configuration files and change the optimization parameters to fine-tune the pre-trained models using the collected data from different LiDAR placements in CARLA. Details of the optimization hyperparameters are given in Table 2. Since only the front half of the point cloud is used to fine-tune the detection models, we change `POINT_CLOUD_RANGE` to be $[0, -20, -3, 40, 20, 1]$ as well. We ensure that the hyperparameters are the same for all experiments to fairly compare the detection performance, and the detection performance is tested at epoch 10 for all models.

2. More Experimental Results and Analysis

2.1. Recall under Different LiDAR Configurations

Apart from the average precision to show the detection performance, we further consider using the overall recall (rcnn) with IoU of 0.5 and 0.7 to show the detection performance of all the objects and validate the relationship be-

Models	Recall rcnn @0.50 IoU				Recall rcnn @0.70 IoU			
	Center	Line	Pyramid	Trapezoid	Center	Line	Pyramid	Trapezoid
PV-RCNN [2]	0.5834	0.6009	0.6260	0.6103	0.4192	0.4330	0.4610	0.4404
Voxel RCNN [1]	0.5686	0.5858	0.6112	0.5929	0.3901	0.4093	0.4299	0.4172
PointRCNN [3]	0.4462	0.4584	0.4722	0.4593	0.3030	0.3346	0.3321	0.3392
PointRCNN-IoU [3]	0.4437	0.4633	0.4706	0.4597	0.3020	0.3300	0.3325	0.3373
SECOND [5]	0.4590	0.4792	0.5035	0.4835	0.2827	0.2960	0.3104	0.3050
SECOND-IoU [5]	0.5788	0.5944	0.6249	0.6052	0.3726	0.3838	0.4107	0.3966

Table 3. Comparison of recall performance under various LiDAR configurations using different algorithms.

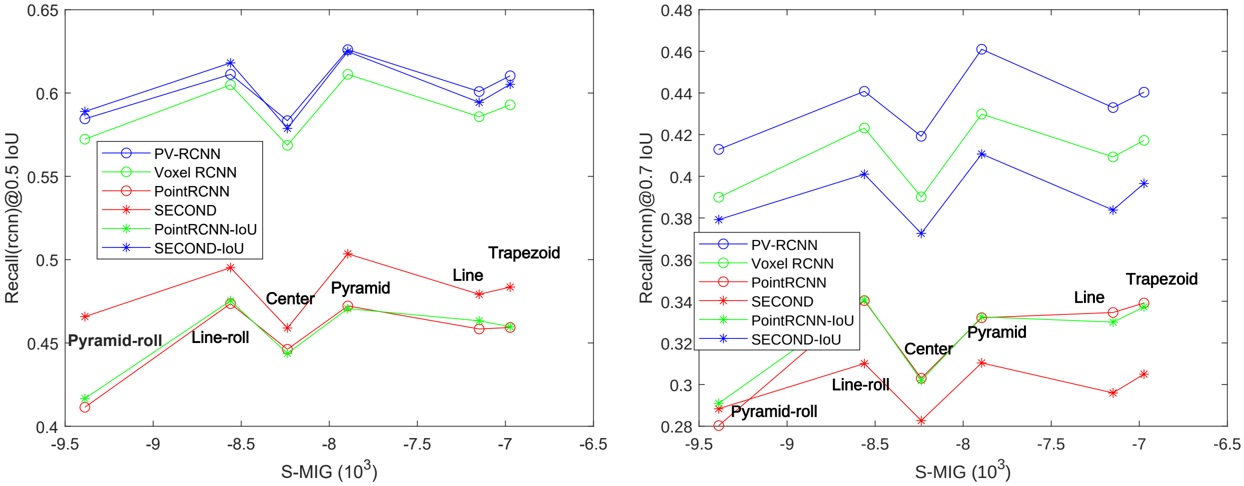


Figure 4. The relationship overall recall performance and overall surrogate metric of different LiDAR placements for Car, Van and Cyclist.

tween our surrogate metric. The comparison of recall performance under different LiDAR placements can be found in Table 3. It can be seen that the recall metric varies a lot under different LiDAR placements for the same detection models. Specifically, *Pyramid* almost gets the best performance for all the algorithms with IoU of both 0.5 and 0.7, which is different from the performance of average precision in Table 1 of the main text of the paper.

From Figure 4, we can find the increasing trend between recall and our surrogate metric as well. Note that since the recall is calculated for all Cars, Vans and Cyclists, our total information gain surrogate metric is the sum of S-MIG of Car, Van and Cyclist. Furthermore, it can be seen that the performance variance under different LiDAR placements does not decrease as the IoU is going less, showing that the influence of LiDAR placement is consistent with the recall metrics, as the surrogate metric shows.

2.2. Sensitiveness Analysis of Detection Algorithms

In this section, we analyze how sensitive current different LiDAR-based detraction algorithms are to the influence of LiDAR placements. From Figure 8 in the main text of the paper and Figure 4 in the supplementary material, it can be found that point-based methods, like *PointRCNN* and *PointRCNN-IoU*, are sensitive to different LiDAR placements and have a relatively clear linear relationship with our surrogate metric. On the contrary, the detection performance of voxel-based methods fluctuates with different LiDAR configurations as well, but the linear relationship is less obvious, which is because point-based methods rely on the original point data collected from LiDAR and are highly related to the point cloud distribution and uncertainty revealed by our surrogate metric.

3D AP	PV-RCNN [2]	PointRCNN [3]	S-MIG (10^3)
Pyramid	57.44	44.28	-5.64
Pyramid-pitch	53.46	37.27	-5.71

BEV AP	PV-RCNN [2]	PointRCNN [3]	S-MIG (10^3)
Pyramid	65.81	56.97	-5.64
Pyramid-pitch	62.33	51.39	-5.71

Table 4. Influence of pitch rotation of front LiDARs on Car AP detection performance.

Besides, there are some detection algorithms where the fluctuation caused by LiDAR placement is even more significant than the difference between different algorithms given any LiDAR placement. Specifically, the recall with 0.7 IoU of *Second* is better than *PointRCNN* under *Pyramid-roll*, while *Second* performs worse than *PointRCNN* using data collected under other LiDAR placements, showing that LiDAR configuration is also a critical factor in object detection. Therefore, there is still room to improve

Recall @0.5 IoU	PV-RCNN [2]	PointRCNN [3]	S-MIG (10^3)
Pyramid	0.6260	0.4722	-7.90
Pyramid-pitch	0.6002	0.4490	-7.99
Recall @0.7 IoU	**PV-RCNN [2]**	**PointRCNN [3]**	**S-MIG (10^3)**
Pyramid	0.4610	0.3321	-7.90
Pyramid-pitch	0.4279	0.3122	-7.99

Table 5. Influence of pitch rotation of front LiDARs on overall recall detection performance.

the 3D detection performance from LiDAR placement.

2.3. Influence of Pitch Angles for Front LiDAR

Along with the analysis of roll angle of sided LiDARs in the main text of the paper, we also show the influence of pitch angle for front LiDAR, which is intuitively essential for object detection in the front 180-degree field of view. From Table 4 and Table 5, it can be seen that the surrogate metric of *Pyramid-pitch* placement is less than that of *Pyramid* placement. The Car average precision and overall recall metrics are less, which shows that our surrogate can be used to evaluate the detection performance around the local neighborhood of LiDAR placement. Note that the surrogate metric is the sum of all the objects for recall comparison because the recall includes all Car, Van and Cyclist.

2.4. LiDAR Placement for Pedestrain Detection

Besides cars, vans and cyclists, e conducted an extra experiment considering the pedestrians mainly from the sidewalk as shown in the table below. The results further consolidate the significance of LiDAR placement to SOTA point cloud-based detection algorithms with the metric of AP at 0.5 IOU. It shows the *Line* option almost performs the best due to its wide horizontal view field. Furthermore, we found that the influence on Pedestrian detection is even larger than Car and Cyclists as shown in Table 1 in the main text of the paper, where the four SOTA models are affected by 30% of *Line* placement.

2.5. Qualitative Visualization and Analysis

This section shows some qualitative visualization and analysis of point cloud collected through different LiDAR placements. From Figure 5, we can see that the distribution of point cloud varies a lot in the same scenario, which is directly caused by different LiDAR placements. Specifically, the point distribution of *Center* is more uniform in the vertical direction. However, there are some aggregated points as clear horizontal lines for other placements, which results in the performance improvement for small object detection like Cyclist or extremely large trucks, as shown in Figure 8 and Table 1 in the main text of the paper.

3D (AP@0.50)	Center	Line	Pyramid	Trapezoid
PV-RCNN [2]	12.80	**16.35**	10.95	13.57
Voxel RCNN [1]	10.95	8.17	4.85	**12.09**
PointRCNN [3]	10.98	**11.75**	10.20	11.32
PointRCNN-IoU [3]	10.66	**12.19**	10.23	11.30
SECOND [5]	**11.35**	8.82	7.46	8.10
SECOND-IoU [5]	5.00	**8.78**	5.23	7.50
BEV (AP@0.50)	Center	Line	Pyramid	Trapezoid
PV-RCNN [2]	15.30	**19.96**	17.27	17.23
Voxel RCNN [1]	11.45	9.32	7.37	**13.48**
PointRCNN [3]	11.86	**14.44**	11.67	12.21
PointRCNN-IoU [3]	11.59	**14.38**	11.85	13.61
SECOND [5]	13.38	**13.71**	10.08	12.57
SECOND-IoU [5]	7.46	**12.27**	8.49	10.54

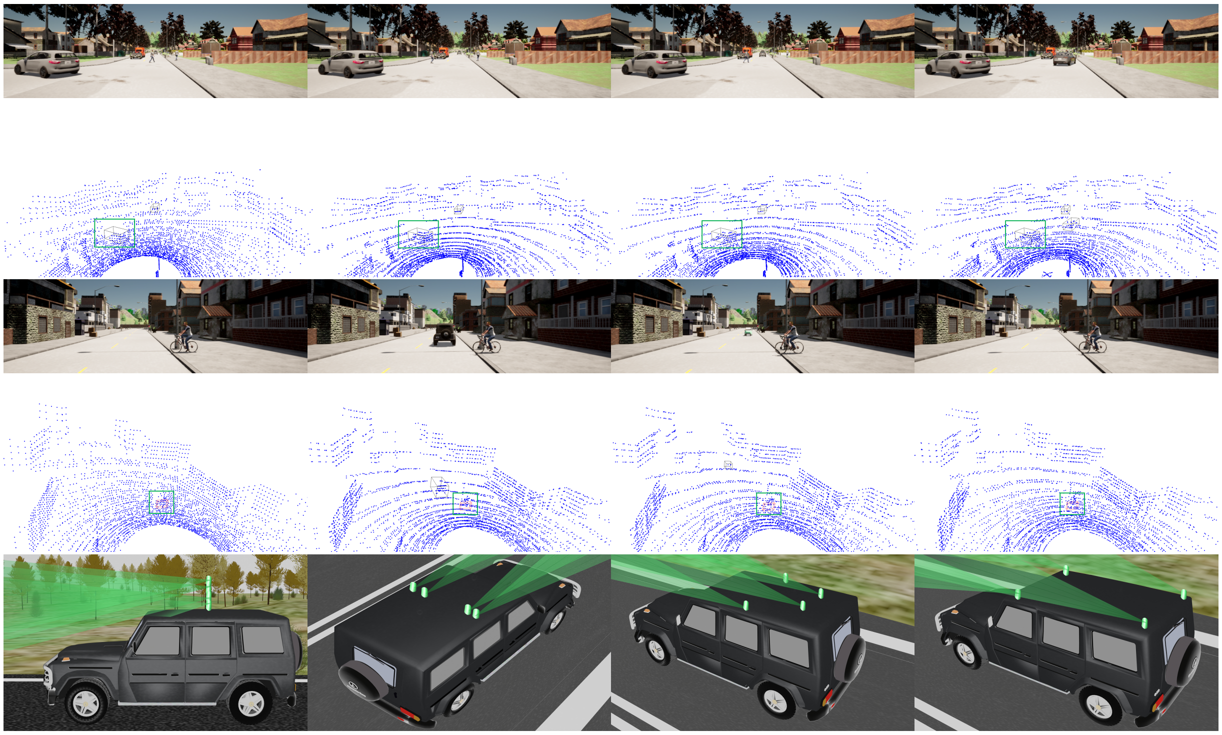
Table 6. Influence of LiDAR placement on pedestrian detection

However, since the aggregated lines can better represent the shape and other critical information of the objects, other plane-placed LiDAR configurations have better performance for large-scale object detection like Car, where there are enough points for the detection task. In Figure 6, we illustrate the influence of roll and pitch angle in *Pyramid* placement. For the objects far away, the distribution of object point cloud is quite different. The distribution under LiDARs with roll angle is sparser horizontally, making key points harder to aggregate. However, *Pyramid-pitch* placement makes the points in the front object shifted below, resulting in the object not being that clear to detect. Note that there is some slight difference in objects for each scenario, but it does not matter because the distribution of objects is the same for all LiDAR placements. As the number of frames is huge, the detection performance will be fair for all the experiments.

3. Limitation and Discussion

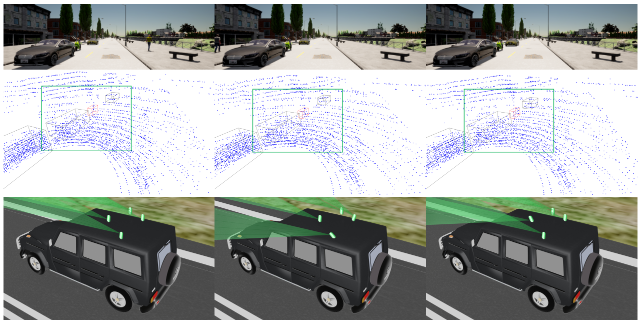
In this section, we present some limitations of our work and some further discussions. First, when calculating our surrogate metric using POG, we have assumed that all the voxels are independent. Although objects in ROI are independent among all the frames, the voxels may be related to their neighbors and not fully independent. However, since the number of frames is large enough, the related voxels are highly separated, so our assumptions somewhat hold. Besides, we do not consider the case of occlusion. When LiDAR beams meet some object, they will no longer get through it in most cases and reflect. In our Bresenham model, we ignore such occlusion based on observing that the occlusion case is rare in the CARLA town with 40, 80, or 120 vehicles. Also, occlusion often happens far away from the ego vehicle, so it may not influence the surrogate metric too much. Also, we leave the task of finding better surrogate functions to evaluate or even optimize LiDAR configurations as the future work in this community.

Although we have conducted extensive experiments and



(a) Center (b) Line (c) Trapezoid (d) Square

Figure 5. Visualization of point cloud distribution from different LiDAR configurations.



(a) Pyramid (b) Pyramid-roll (c) Pyramid-pitch

Figure 6. Comparison of point cloud distribution from LiDAR placement with roll and pitch angles.

made comprehensive comparison and analyses, there are still some weaknesses. First, we do not train each model for 80 epochs as default to make the experiment efficient. We have found that the fine-tuning loss of all the detection algorithms has already converged after 10 epochs, so we stop it and test the performance for a fair evaluation. Also, the point cloud data is sometimes sparse for objects far away, which is challenging for detection and the detection metrics are not as high as those on KITTI. That is based on the observation that the influence of LiDAR placement is more evident for sparse point cloud to avoid saturation and is also consistent with the practical applications where multiple LiDARs are mainly used towards the challenging cases to detect objects with sparse points. Moreover, the evaluated multi-LiDAR placements are not as complicated as the cases in the company. The real LiDAR placement is confidential for the company, and we believe the simplified case is enough to show the influence of LiDAR configuration to meet the motivation of this work.

References

[1] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. *arXiv preprint arXiv:2012.15712*, 2020. 3, 4

[2] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020. 3, 4

[3] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 3, 4

[4] OpenPCDet Development Team. Openpcdet: An open-source toolbox for 3d object detection from point clouds. https://github.com/open-mmlab/OpenPCDet, 2020. 1

[5] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 3, 4