Supplementary Material for "PG-RCNN: Semantic Surface Point Generation for 3D Object Detection"

Inyong Koo* Inyoung Lee* Se-Ho Kim Hee-Seon Kim Woo-jin Jeon Changick Kim

KAIST
Daejeon, South Korea
{iykoo010, inzero24, ksh1040, hskim98, woojin.jeon337, changick}@kaist.ac.kr

S1. Data and Code License

We used the KITTI dataset [2] and the Waymo Open Dataset [7] for our experiments. Both datasets are licensed and widely used for academic research.

Our code is licensed under the "Apache License 2.0". We have included our code and a trained checkpoint of PG-RCNN in the supplementary material, and it will be released as a Github repository after the conference. Please refer to the `README.md` file in the supplementary material for details.

S2. Details of Training Losses

In Section 3.4 of the submitted manuscript, we provide an explanation of the training losses for PG-RCNN, with a focus on the novel point generation loss $\mathcal{L}\mathrm{RPG}$. In this section, we provide further details on the conventional training losses used for two-stage detectors, specifically the region proposal loss $\mathcal{L}\mathrm{RPN}$ and the proposal refinement loss $\mathcal{L}_{\mathrm{head}}$.

Our implementation of $\mathcal{L}_{\mathrm{RPN}}$ is consistent with those used in prior works such as [8, 4, 6, 1]. For dense predictions generated by the proposal layers, we assign classification targets based on IoU thresholds for each class, as follows:

$$c_i^* = \begin{cases} c & \mathrm{IoU}_i^c \geq \theta_H^c, \\ 0 & \mathrm{IoU}_i^c < \theta_L^c, \end{cases} \tag{S1}$$

where IoU_i^c is the IoU between the i-th anchor proposal and the corresponding ground truth bounding box with class label c, and θ_H^c and θ_L^c represent the foreground and background IoU thresholds, respectively, for class c. To calculate the classification term of the region proposal loss $\mathcal{L}_{\mathrm{RPN}-cls}$, we sample 512 anchors and apply Focal Loss [10]:

$$\mathcal{L}_{\mathrm{RPN}-cls}(p_i^a, c_i^*) = -\alpha(1 - p_i^a(c_i^*))^\gamma \log(p_i^a(c_i^*)). \tag{S2}$$

Here, $p_i^a(c^i)$ denotes the i-th anchor proposal's classification branch output for the class probability of c^i. The parameters of the focal loss, α and γ, are set to $\alpha = 0.25$ and $\gamma = 2$ for our training process.

The regression target is obtained using the residuals between the anchor box and the ground truth bounding box. A bounding box is represented with seven parameters $b = (x, y, z, l, w, h, \theta)$, where x, y, and z are the center coordinates, w, l, and h are the width, length, and height, respectively, and θ is the yaw rotation around the z-axis. The residuals are calculated as follows:

$$\begin{aligned} &\Delta x = \frac{x^{gt} - x^a}{d^a},\ \Delta y = \frac{y^{gt} - x^a}{d^a},\ \Delta z = \frac{z^{gt} - z^a}{h^a}, \\ &\Delta w = \log\frac{w^{gt}}{w^a},\ \Delta l = \log\frac{l^{gt}}{l^a},\ \Delta h = \log\frac{h^{gt}}{h^a}, \\ &\Delta\theta = \sin(\theta^{gt} - \theta^a), \end{aligned} \tag{S3}$$

where the parameters for ground truth and anchor boxes are indicated with the superscripts gt and a, respectively, and $d^a = \sqrt{(l^a)^2 + (w^a)^2}$ is the diagonal of the base of the anchor box. Note that in our definition of $\Delta\theta$, θ^a and $\theta^a + \pi$ yield the same residual. To address the direction ambiguity, we incorporate a direction classifier into the regression branch, which is trained using a cross-entropy loss function. The direction classifier target $\delta^*(d)$ is positive when the $\theta > 0$ and negative otherwise. The regression term of the region proposal loss $\mathcal{L}_{\mathrm{RPN}-reg}$ is composed of the Smooth-L1 loss for the bounding box parameters and the

*Denote equal contribution

cross-entropy loss for the direction classifier. *i.e.*,

$$\begin{aligned}\mathcal{L}_{\mathrm{RPN}-reg}(\delta_i^a,\delta_i^*) = \\ &\sum_{b\in\{x,y,z,l,w,h,\theta\}} SmoothL1(\delta_i^a(b),\Delta b) \\ &+\beta_{dir} CrossEntropy(\delta_i^a(d),\delta_i^*(d)),\end{aligned} \tag{S4}$$

where δ_i^a is the output of the regression branch, and $\beta_{dir} = 0.1$ is a balancing parameter for the direction classifier loss. The total region proposal loss $\mathcal{L}_{\mathrm{RPN}}$ is composed of

$$\begin{aligned}\mathcal{L}_{\mathrm{RPN}} = \frac{1}{N_{fa}}\sum_i [\mathcal{L}_{\mathrm{RPN}-cls}(c_i^a,c_i^*) \\ +\mathbb{1}(c_i^* \geq 1)\beta_{reg}\mathcal{L}_{\mathrm{RPN}-reg}(\delta_i^a,\delta_i^*)]\,,\end{aligned} \tag{S5}$$

where N_{fa} is the number of foreground anchors, $\beta_{reg} = 2$ is a balancing parameter for regression loss, and $\mathbb{1}(c_i^* \geq 1)$ indicates that only foreground anchors contribute to the regression loss.

Similarly, the proposal refinement loss $\mathcal{L}_{\mathrm{head}}$ is calculated with the detection head outputs as follows:

$$\begin{aligned}\mathcal{L}_{\mathrm{head}} = \frac{1}{N_{sp}}\sum_i [\mathcal{L}_{\mathrm{head}-cls}(l_i^p,l_i^*) \\ +\mathbb{1}(\mathrm{IoU}_i \geq \theta_{reg})\mathcal{L}_{\mathrm{head}-reg}(\delta_i^p,\delta_i^*)]\,,\end{aligned} \tag{S6}$$

where N_{sp} is the number of sampled proposals, and $\mathcal{L}_{\mathrm{head}-cls}$ and $\mathcal{L}_{\mathrm{head}-reg}$ are the classification and regression loss terms acquired from the confidence branch and refinement branch, respectively. $\mathbb{1}(\mathrm{IoU}_i \geq \theta_{reg})$ indicates that the regression loss is only calculated with proposals with IoU over a threshold θ_{reg}. The classification target assigned for detection head loss l_i^* is an IoU-related value, similar to Eq. S1

$$l_i^* = \begin{cases} 1 & \mathrm{IoU}_i \geq \theta_H, \\ \frac{\mathrm{IoU}_i-\theta_L}{\theta_H-\theta_L} & \theta_L \leq \mathrm{IoU}_i < \theta_H, \\ 0 & \mathrm{IoU}_i < \theta_L. \end{cases} \tag{S7}$$

Here, we used the binary cross-entropy loss for $\mathcal{L}_{\mathrm{head}-cls}$. The regression targets for the detection head are similarly obtained as in Eq. S3. We don't attach a direction classifier here. Instead, we calculate the corner points of the given and ground-truth bounding boxes and use the residual to further regularize the regression process. The regression term of the proposal refinement loss $\mathcal{L}_{\mathrm{head}-reg}$ is comprised of:

$$\begin{aligned}\mathcal{L}_{\mathrm{RPN}-reg}(\delta_i^p,\delta_i^*) = \\ &\sum_{b\in\{x,y,z,l,w,h,\theta\}} SmoothL1(\delta_i^p(b),\Delta b) \\ &+\sum_{j=1,2,\cdots,8} SmoothL1(\delta_i^p(c_j),\delta_i^*(c_j)),\end{aligned} \tag{S8}$$

where $\delta_i^p(c_j)$ and $\delta_i^*(c_j), j = 1, 2, \cdots, 8$ are the eight corner point coordinates of i-th proposal and its corresponding ground truth bounding box.

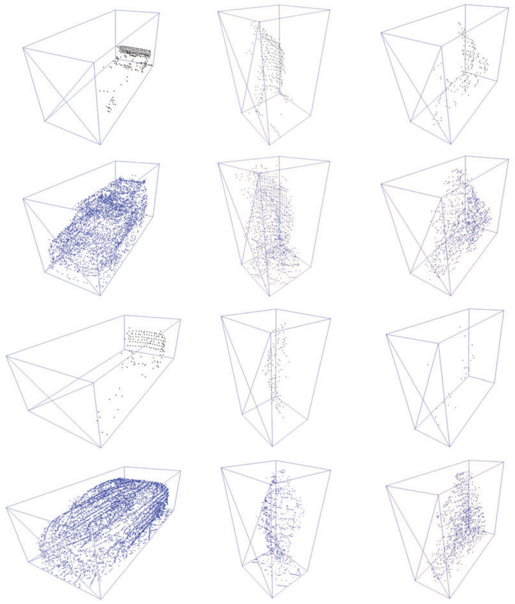

Figure S1. Examples of completed point clouds for cars, pedestrians, and cyclists in Waymo Open Dataset.

Table S1. Performance comparison on the validation set of Waymo Open Dataset. All training data was used for this experiment. The best performance value is in **bold**.

	Method	Vehicle 3D AP$_{R40}$			
		Overall	0-30m	30-50m	50m-Inf
L1	PV-RCNN [6]	70.30	91.92	69.21	42.17
	Voxel R-CNN [1]	75.59	**92.49**	74.09	**53.15**
	PG-RCNN (Ours)	**75.70**	91.46	**74.14**	52.87
L2	PV-RCNN [6]	65.36	91.58	65.13	36.46
	Voxel R-CNN [1]	66.59	**91.74**	**67.89**	40.80
	PG-RCNN (Ours)	**67.36**	90.23	67.76	**40.98**

S3. Experiments on Waymo

S3.1. Experimental Setup

Waymo Open Dataset. The Waymo Open Dataset [7] is a large-scale autonomous driving dataset containing 798 training sequences (around 158k point cloud samples) and 202 validation sequences (around 40k point cloud samples).

Our method requires complete point clouds of the objects to supervise point generation. To accomplish this, we approximated the complete shape by utilizing different instances of the same object class. In the KITTI dataset [2], we searched for objects that display similar point distributions and bounding boxes and combined their point clouds with the original point cloud to create a dense point

Table S2. Performance comparison on the validation set of Waymo Open Dataset. 20% of training data was used for this experiment. The best performance value is in **bold**.

Method	Vehicle				Pedestrian				Cyclist			
	LEVEL_1		LEVEL_2		LEVEL_1		LEVEL_2		LEVEL_1		LEVEL_2	
	mAP	mAPH	mAP	mAPH	mAP	mAPH	mAP	mAPH	mAP	mAPH	mAP	mAPH
PV-RCNN [6]	**75.29**	**74.61**	**66.75**	**66.12**	72.12	60.71	63.05	52.90	66.43	64.39	63.97	62.00
Voxel R-CNN [1]	74.82	74.23	66.16	65.62	66.66	57.43	57.74	49.61	68.57	67.01	66.03	64.53
PG-RCNN (Ours)	74.57	74.09	66.16	65.71	**77.06**	**70.90**	**68.41**	**62.73**	**69.76**	**68.65**	**67.16**	**66.10**

cloud. However, the Waymo Open Dataset contains significantly more object instances than KITTI (# cars: 1.02M vs. 10.7K), which presents a challenge for using the complete shape approximation method as in KITTI. Fortunately, objects in Waymo Open Dataset often reappear in other point cloud samples that belong to the same sequence, and the object instance IDs are provided. We can track the object instance and use the object point clouds from different views to augment the original point cloud. For vehicles and cyclists, we mirror the points along the object's heading axis, assuming symmetry, as we did in the KITTI dataset. Figure S1 displays examples of the completed point clouds used for generation targets. Note that we were not able to provide generation targets for all annotated objects due to insufficient collected points. For example, we managed to produce dense generate targets for 31,126 unique car objects but missed 15,296. In future work, a better data preparation method could be explored, such as using a point completion network [9] trained with available data to produce dense complete point clouds for all objects.

Implementation Details. The detection range for Waymo Open Dataset is [-75.2m, 75.2m], [-75.2m, 75.2m], and [-2m, 4m] for the X, Y, and Z-axis respectively, and a voxel size of (0.1m, 0.1m, 0.15m) is used. We used an almost identical network architecture in KITTI for the experiments, except using an increased number of channels of the proposal layers to (128, 256) and 192 for grid feature dimension.

PG-RCNN is trained using the Adam optimizer [3] with a one-cycle policy for 30 epochs with an initial learning rate of 0.01. We calculate the point-level segmentation loss $\mathcal{L}_{score}$ on 4,096 points. Due to a lack of time and resources, we did not fully tune the hyperparameters for the Waymo Open Dataset. We expect that employing a more sophisticated detection head can improve scalability to larger datasets. We will report a better configuration for Waymo Open Dataset on our code release.

Experiment results We compare the performance of our method with two significant works, PV-RCNN [6], and Voxel R-CNN [1]. The model trained on the training set is evaluated on the validation set using mean average precision (mAP) and mean average precision weighted by heading (mAPH), using IoU thresholds of 0.7 for Vehicles and 0.5 for Pedestrians and Cyclists. Objects are categorized into two groups based on the number of points within their bounding boxes: LEVEL_1 (L1) includes more than five points, while LEVEL_2 (L2) includes fewer than five points.

Table S2 shows the 3D detection results for vehicles, pedestrians, and cyclists. Here, we re-implemented the previous methods and compared the performance with ours, using 20% of training data. Without bells and whistles, our PG-RCNN shows comparable performance with the two previous methods. Especially, PG-RCNN significantly outperforms previous methods for pedestrians and cyclists.

Table S1 shows the 3D detection performance on vehicles in different distance ranges, using all training data. The results for PV-RCNN and Voxel R-CNN are obtained from their papers. PG-RCNN achieves a better performance overall, but we admit that our method does not clearly outperform previous methods in the experiments. Nevertheless, the qualitative results (Fig. S2) show promising detection and point cloud generation performance. We will continue evaluations on Waymo Open Dataset with different configurations and data preparation methods to report better settings for the dataset.

S4. More Qualitative Results

In our main paper, we only showed the visualization for car detection results to compare with SIENet [5]. Figure S3 shows more examples on the KITTI dataset that includes detection results of pedestrians and cyclists. The qualitative results show that PG-RCNN is also capable of generating accurate point clouds for pedestrians and cyclists.

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. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1201–1209, 2021.

[2] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012.

[3] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun,

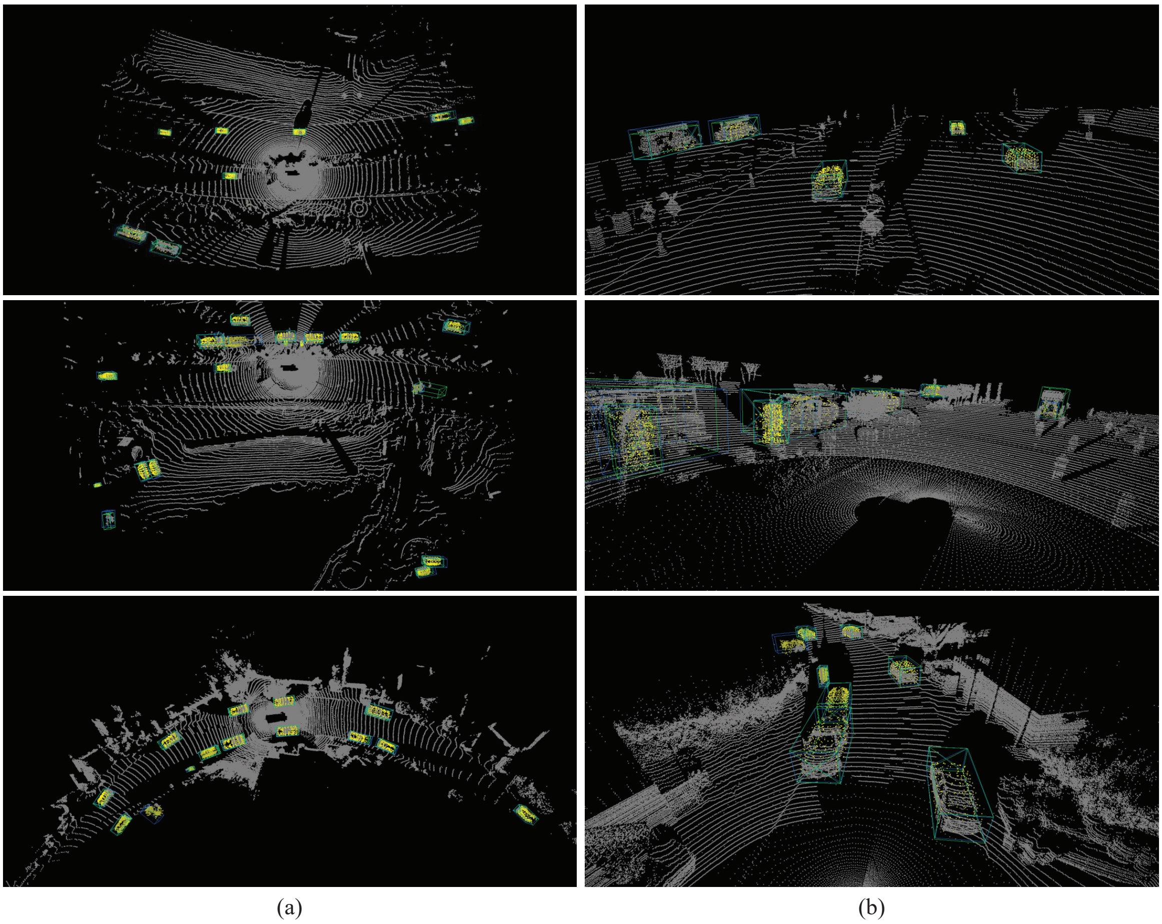

Figure S2. Point generation and detection results on Waymo Open Dataset. (a) Bird-eye-view and (b) zoom-in view. The generated points, predicted bounding boxes, and ground truth bounding boxes are highlighted in yellow, green, and blue, respectively.

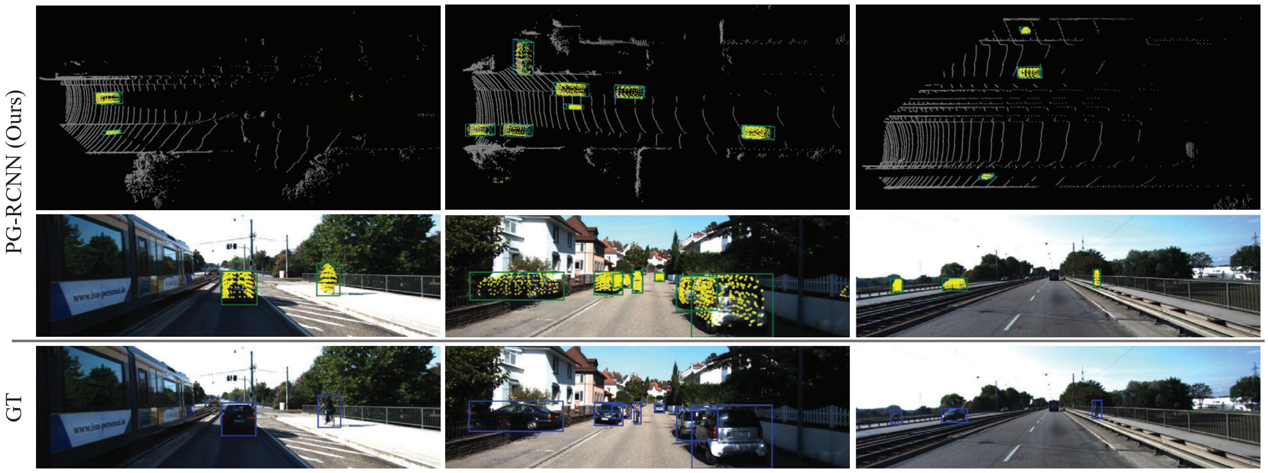

Figure S3. More qualitative results on KITTI dataset. The generated points, predicted bounding boxes, and ground truth bounding boxes are highlighted in yellow, green, and blue, respectively.

editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[4] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12697–12705, 2019.

[5] Ziyu Li, Yuncong Yao, Zhibin Quan, Jin Xie, and Wankou Yang. Spatial information enhancement network for 3d object detection from point cloud. *Pattern Recognition*, 128:108684, 2022.

[6] 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.

[7] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020.

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

[9] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *2018 international conference on 3D vision (3DV)*, pages 728–737. IEEE, 2018.

[10] Peng Yun, Lei Tai, Yuan Wang, Chengju Liu, and Ming Liu. Focal loss in 3d object detection. *IEEE Robotics and Automation Letters*, 4(2):1263–1270, 2019.