Supplementary Material: Generative Range Imaging for Learning Scene Priors of 3D LiDAR Data

Kazuto Nakashima[1] Yumi Iwashita[2] Ryo Kurazume[1]
[1]Kyushu University, Fukuoka, Japan
[2]Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA
k_nakashima@irvs.ait.kyushu-u.ac.jp yumi.iwashita@jpl.nasa.gov kurazume@ait.kyushu-u.ac.jp

1. Overview

This supplementary material summarizes implementation details of our model architectures and experiments in Section 2, detailed analysis of evaluation metrics in Section 3, generated examples of our method and baselines in Section 4, and Sim2Real semantic segmentation results in Section 5.

2. Implementation details

2.1. Models

Fig. 1 shows an overview of our proposed GAN framework. We design the generator network based on INR-GAN [16], which was proposed to generate natural images in coordinate-based representation.

Generator. The generator is composed of a mapping network and synthesis blocks as shown in Fig. 1a. The mapping network transforms the latent space $z \sim N(0, I)$ into another representation, style space w, which modulates the weights of the synthesis blocks Ω. The synthesis blocks represent the function which returns inverse depth x_d and ray-drop probability x_n given the specific angles $\Phi = (\theta, \phi)$. The outputs x_d and x_n are then converted to the final LiDAR image x_G through the lossy measurement model. Each synthesis block encodes the angular inputs to high-dimensional space to represent spatial bias using Fourier features [17]. Note that all operations in synthesis blocks are pixel-independent while the set of angles Φ is dowmsampled hierarchically to perform with a reasonable computational cost as proposed in INR-GAN.

Discriminator. For the discriminator in Fig. 1c, we use the same setup of DUSty [11] while replace the backbone with StyleGAN2 [10]. We applied the separable blur filter [8] to the discriminator inputs and modify all the kernels with circular padding.

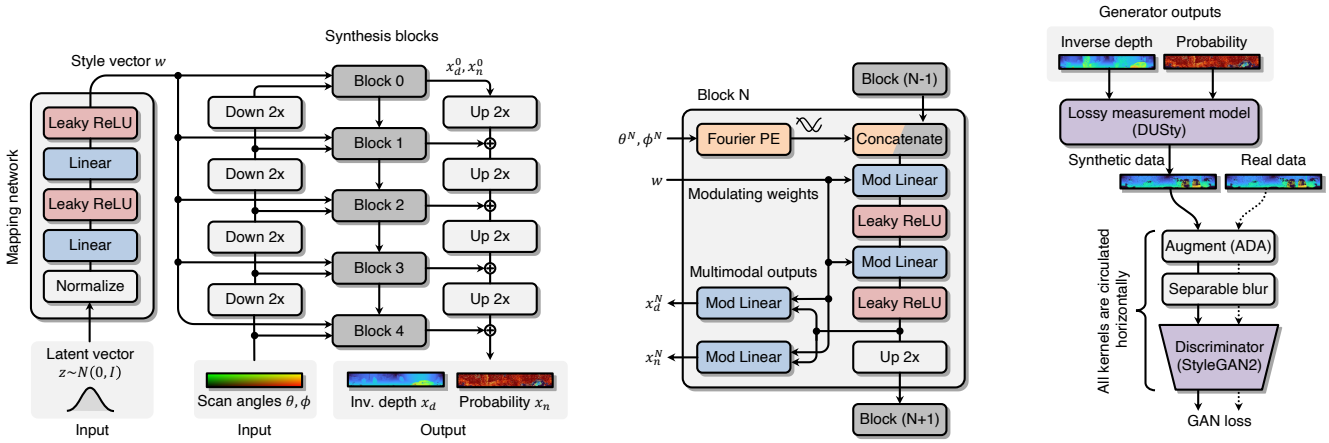
2.2. Training

We employed the adaptive discriminator augmentation (ADA) [9] for all the image-based methods: vanilla GAN, DUSty, and ours. The augmentation basically followed the original pipeline by Karras *et al.* [9], but disabled the steps of rotation and horizontal scaling that break the circular structure of range images. We also modified the integer/fractional translation into circulating behavior. We believe that it is required to explore the optimal augmentation set for LiDAR range images, while the tuning remains for future work.

As the adversarial objective, we employed the non-saturating loss with a gradient penalty [10]. The penalty coefficient was set to 1. All parameters were updated by Adam optimizer for 25M iterations with a learning rate of 0.002 and a batch size of 48. Training were performed on three NVIDIA RTX 3090 GPUs.

2.3. Computational cost of EMD

Earth mover's distance (EMD) is one of the metrics measuring the error between point clouds. Compared to the other metrics such as chamfer distance, EMD reflects the local details and the density distribution and is popular for the assessment of point clouds. However, it is known that computing EMD has an $o(N^3)$ complexity where N is the number of points in 3D point clouds [12]. This is problematic for our case using LiDAR point clouds, for instance, in training point-based models such as l-WGAN [1] and computing the standard evaluation metrics such as COV, MMD, and 1-NNA [20]. In Fig. 2, we compute a pairwise distance of $M = 10,000$ sets of N points, where N ranges from 2^9 to 2^{13} with a batch size of 256, and show the computation time as a function of the number of points. Similar to Nakashima *et al.* [11], we reduce the number of points to conventional 2048 by farthest point sampling in conducting experiments with point-based methods and evaluating the point-based metrics.



(a) An overview of our generator (b) A detail of synthesis block (c) Discriminator

Figure 1: Building blocks of our proposed GAN framework.

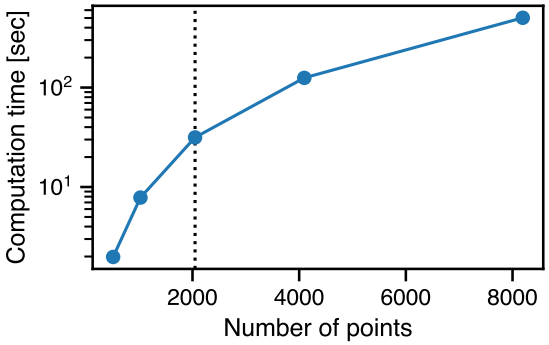


Figure 2: EMD computation time as a function of the number of points. The conventional number of point cloud tasks is 2048 (dotted line), while our task uses $64 \times 512 = 32,768$ points in full setting.

2.4. Inference

For the inference application, we use the style code w instead of the latent code z to gain reconstruction fidelity as demonstrated in the related studies [10, 16, 2, 9, 14]. We optimize the style code w for 500 iterations in the first step (GAN inversion) and then optimize the generator weights Ω for another 500 iterations for the second step (pivotal tuning). We empirically set the learning rate for 0.05 and 0.0005 for the first and second steps, respectively.

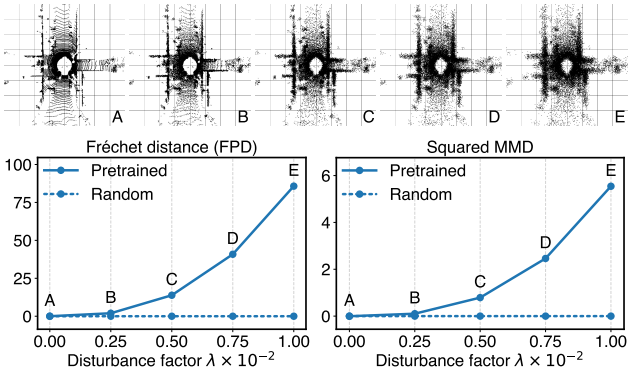
3. Sanity check of evaluation metrics

For evaluating GANs, we used two types of distributional metrics on the PointNet representation: Fréchet distance [15] (named FPD for point clouds), squared maximum mean discrepancy (squared MMD) [3]. This section aims to verify if the metrics can be used for evaluating LiDAR point clouds, since the metrics have been designed for other domains. For instance, FPD [15] has been proposed for evaluating ShepeNet [5] generation task where each sample forms small-scale point clouds uniformly sampled from CAD objects. Squared MMD [3] was used to extend Fréchet Inception distance (FID) [7] that is the standard metrics for an image generation task. In the image domain, the metrics are known as Kernel Inception distance (KID) in tribute to the Inception feature extractor. For the backbone of the feature extractor, we used the off-the-shelf PointNet [13] provided by Shu *et al.* [15]. The PointNet backbone[1] is pre-trained on the ShapeNet dataset and used by the original FPD [15]. To verify if the score is derived from learned features or architecture bias, we compute the metrics using two PointNet encoders with pre-trained weights and random weights. All metrics are computed between clean and disturbed sets of KITTI point clouds. In Fig. 3, we provide the results under six types of disturbances; (a) additive Gaussian noises, (b) drop-in Gaussian noises, (c) inflating coordinates, (d) yaw rotation, and (e,f) translation in x/y directions. From the results, we can see that both metrics reflect the distributional error if using the pre-trained PointNet. We can also see that the metrics sensitive to the translation changes in Fig. 3c–f. Although there are scale gaps depending on the type of disturbance, the results are roughly similar to the sanity check of FID [7]. Therefore, we concluded that the two metrics can be used to evaluate the generative models on LiDAR point clouds.

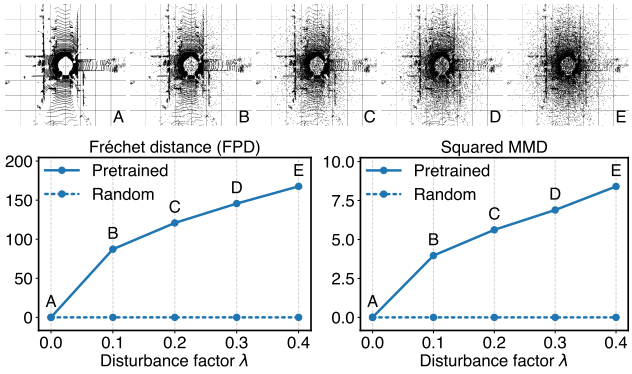
4. Generated examples

In Fig. 4, we provide uncurated sets of real and generated samples from image-based methods including ours. Fig. 5

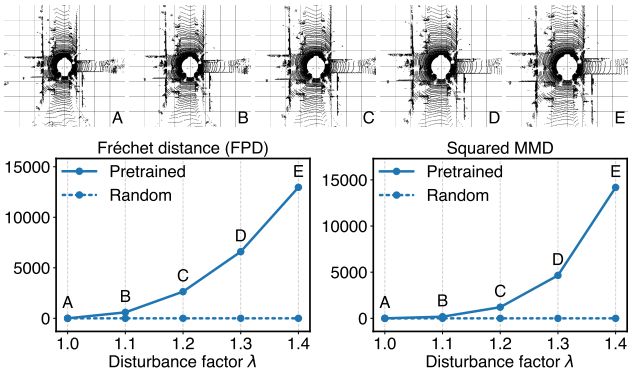
[1] https://github.com/seowok/TreeGAN



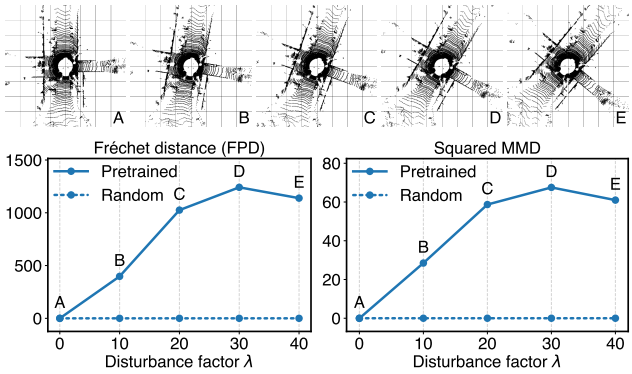
(a) Additive Gaussian noises with a coefficient λ



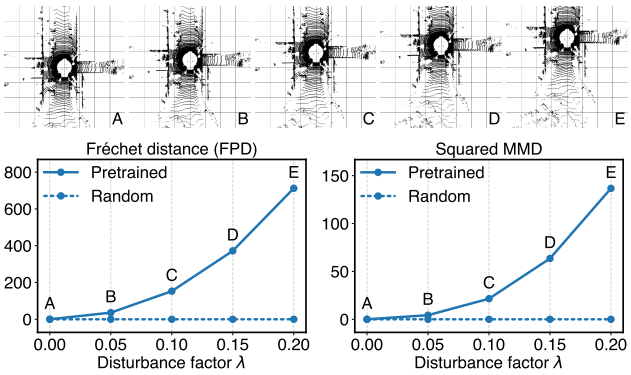
(b) Drop-in Gaussian noises for $\lambda \times 100$ (%) of points



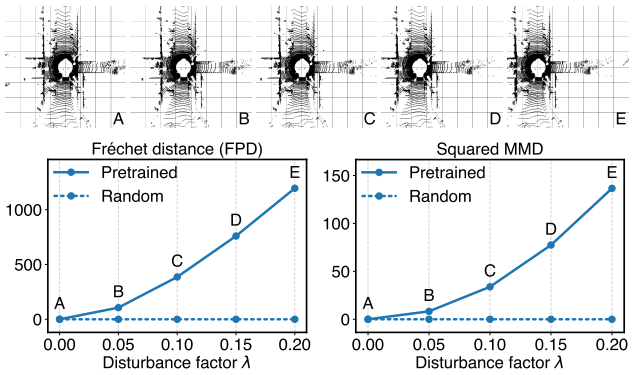
(c) Inflating coordinates with a multiplicative factor λ



(d) Clockwise yaw rotation with an angle λ (°)

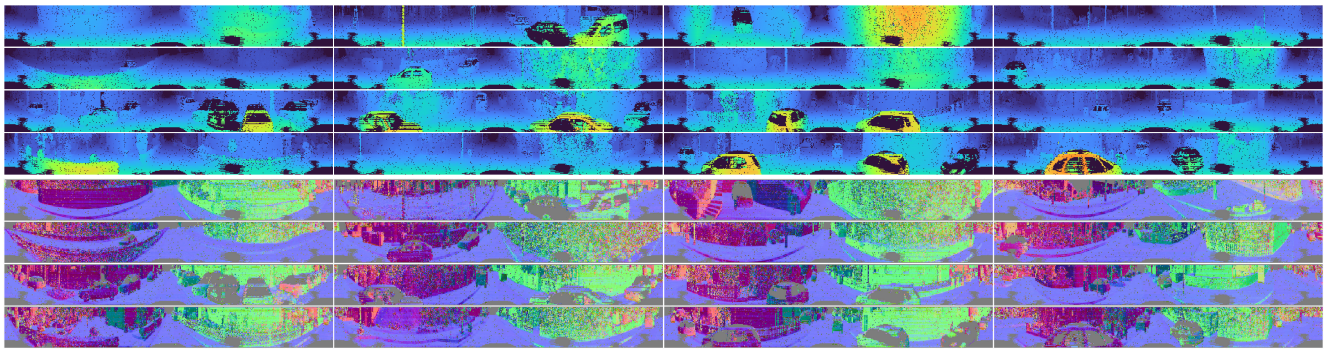


(e) Translation in x direction by λ

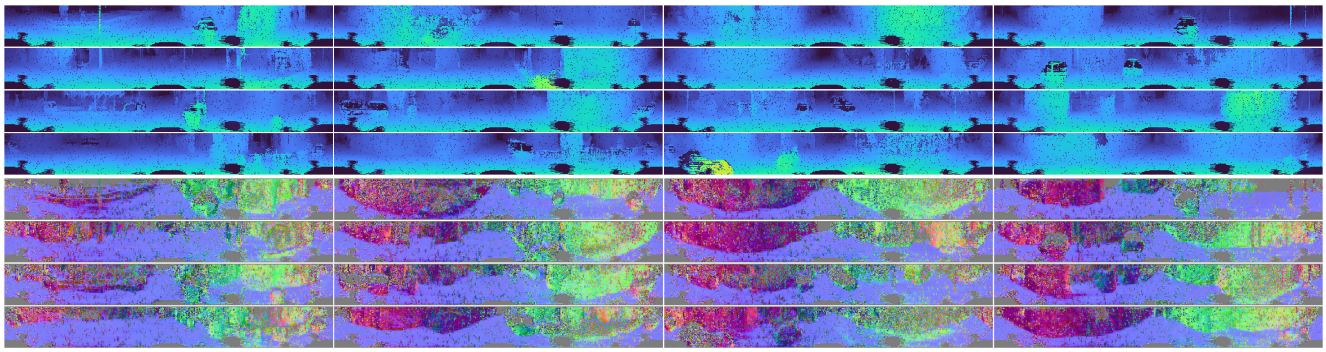


(f) Translation in y direction by λ

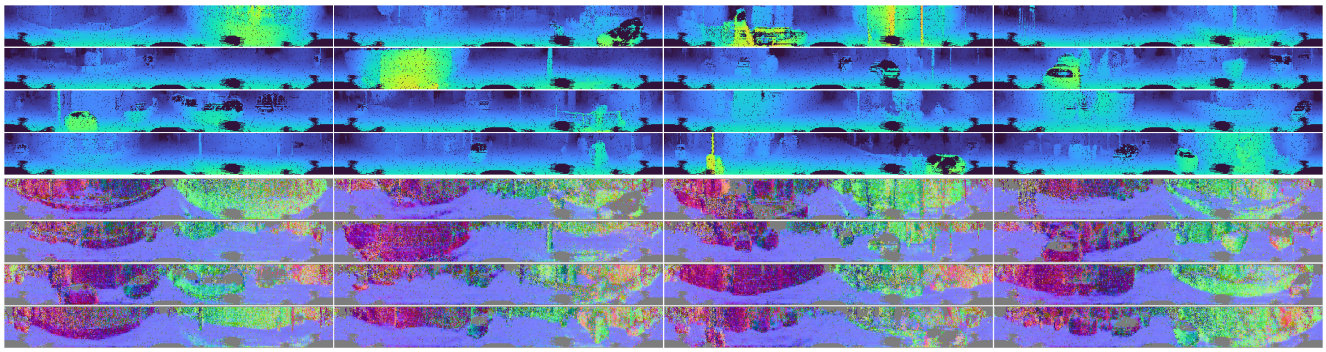
Figure 3: Disturbance sensitivity of four metrics: FPD [15] (Fréchet distance for point clouds) and squared maximum mean discrepancy (squared MMD) [3]. We applied six types of disturbances to the KITTI point clouds with various strength (see A–E) and computed the metrics with the *clean* original point clouds. All point clouds were encoded by PointNet [13] with pre-trained or random weights.



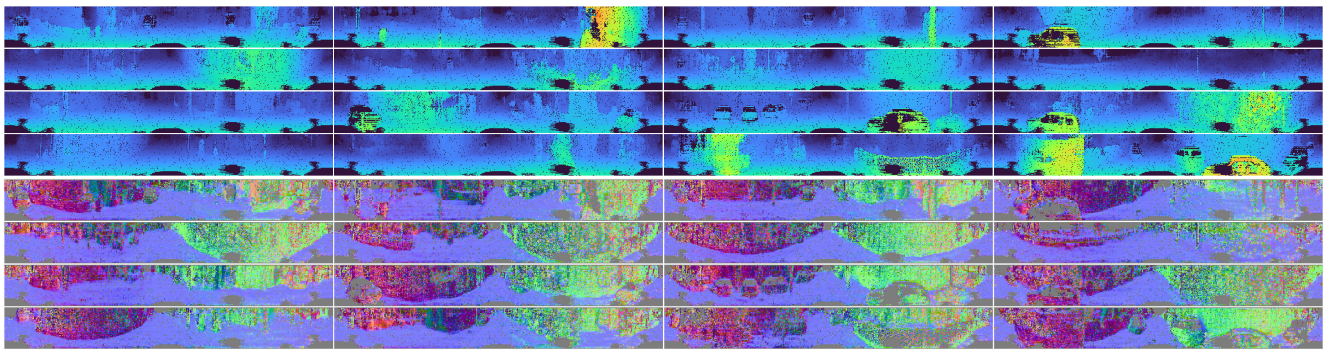
Real data



Vanilla GAN [4]



DUSty [11]



Ours

Figure 4: Qualitative comparison of uncurated sets of generated samples in the image format (top) and the corresponding surface normal maps (bottom). The surface normal maps are computed from projected Cartesian points.

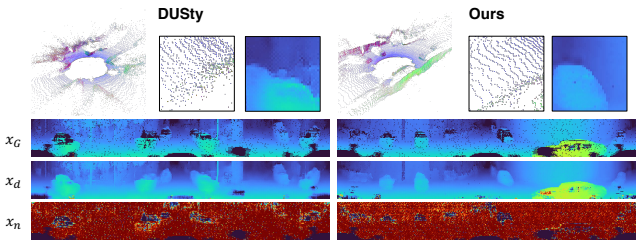


Figure 5: Qualitative comparison between DUSty [11] and ours. From top to bottom: generated point clouds, the final inverse depth maps x_G, the complete depth maps x_d, and the ray-drop probability maps x_n.

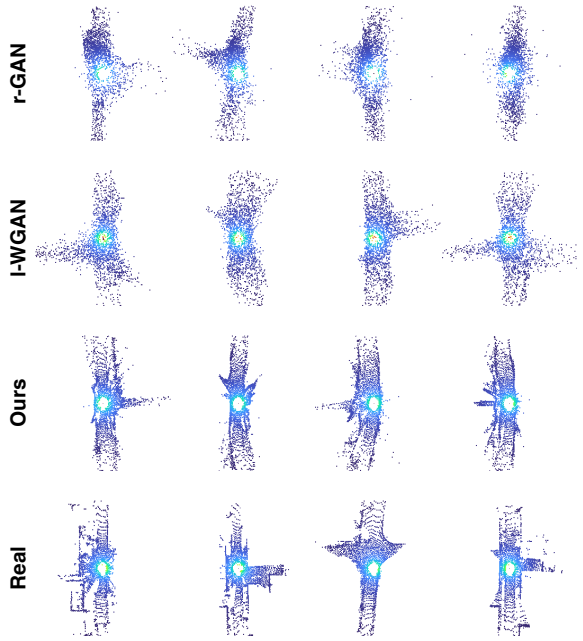


Figure 6: Qualitative comparison in bird's eye views of real and generated point clouds.

compares the results between ours and the most closely related work, DUSty [11]. A close-up comparison shows that baseline methods include checkerboard artifacts and our method succeeded in expressing the smooth road surface. In Fig. 6, we provide uncurated sets of real and generated samples from point-based methods and ours. Our method is superior in point density distribution and edges. In Fig. 7, we show reconstruction examples by our auto-decoding method. From the real data via the lossy measurement, our model produced the smooth shapes and the reasonable ray-drop probability maps. For instance, the ray-drop probabilities have uncertainty on the object edges.

5. Sim2Real semantic segmentation

In Fig. 8, we show Sim2Real segmentation results on KITTI-frontal [19]. All models are trained on GTA-LiDAR while the ray-drop priors are different. We can see that our method (config-E) greatly improved the false negative regions of car classes.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 40–49, 2018.

[2] Ivan Anokhin, Kirill Demochkin, Taras Khakhulin, Gleb Sterkin, Victor Lempitsky, and Denis Korzhenkov. Image generators with conditionally-independent pixel synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14278–14287, 2021.

[3] Mikołaj Bińkowski, Dougal J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[4] Lucas Caccia, Herke van Hoof, Aaron Courville, and Joelle Pineau. Deep generative modeling of lidar data. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5034–5040, 2019.

[5] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An information-rich 3D model repository. Technical report, 2015.

[6] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research (IJRR)*, 32(11):1231–1237, 2013.

[7] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

[8] Takuhiro Kaneko and Tatsuya Harada. Noise robust generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8404–8414, 2020.

[9] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 12104–12114, 2020.

[10] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8110–8119, 2020.

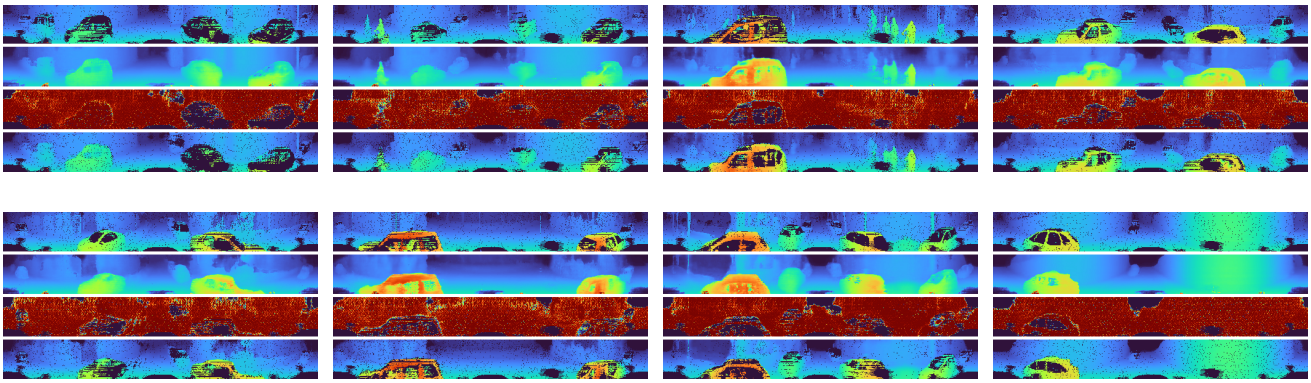


Figure 7: Reconstruction examples by our auto-decoding method. For each group, from top to bottom, we show the target range image $\hat{x}$ from KITTI [6], the generated inverse depth map x_d, the generated ray-drop probability map x_n, and the final output x_G.

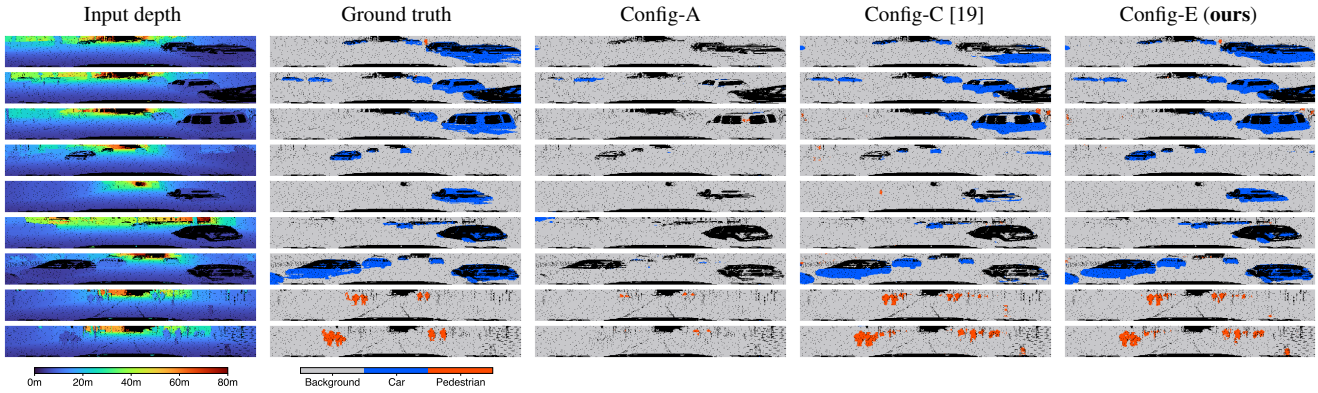


Figure 8: Comparison of Sim2Real segmentation results in 2D range images. We compare three types of ray-drop priors from our main paper. Config-A: GTA-LiDAR without ray-drop rendering. Config-C: GTA-LiDAR with Bernoulli noises from the pixel-wise frequency [18]. Config-E (ours): GTA-LiDAR with Bernoulli noises from our auto-decoded ray-drop probability.

[11] Kazuto Nakashima and Ryo Kurazume. Learning to drop points for lidar scan synthesis. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 222–229, 2021.

[12] Ofir Pele and Michael Werman. Fast and robust earth mover's distances. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 460–467, 2009.

[13] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[14] Daniel Roich, Ron Mokady, Amit H. Bermano, and Daniel Cohen-Or. Pivotal tuning for latent-based editing of real images. *ACM Transactions on Graphics (TOG)*, 42(1), 2022.

[15] Dong Wook Shu, Sung Woo Park, and Junseok Kwon. 3d point cloud generative adversarial network based on tree structured graph convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[16] Ivan Skorokhodov, Savva Ignatyev, and Mohamed Elhoseiny. Adversarial generation of continuous images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10753–10764, 2021.

[17] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 7537–7547, 2020.

[18] Bichen Wu, Alvin Wan, Xiangyu Yue, and Kurt Keutzer. SqueezeSeg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1887–1893, 2018.

[19] Bichen Wu, Xuanyu Zhou, Sicheng Zhao, Xiangyu Yue, and Kurt Keutzer. SqueezeSegV2: Improved model structure and unsupervised domain adaptation for road-object segmenta-

tion from a lidar point cloud. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4376–4382, 2019.

[20] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. PointFlow: 3d point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4541–4550, 2019.