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

Nakashima, Kazuto
Kyushu University

Iwashita, Yumi
Jet Propulsion Laboratory, California Institute of Technology

Kurazume, Ryo
Kyushu University

出版情報：IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp.1256-1266, 2023. Institute of Electrical and Electronics Engineers (IEEE)
バージョン：

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

Abstract

3D LiDAR sensors are indispensable for the robust vision of autonomous mobile robots. However, deploying LiDAR-based perception algorithms often fails due to a domain gap from the training environment, such as inconsistent angular resolution and missing properties. Existing studies have tackled the issue by learning inter-domain mapping, while the transferability is constrained by the training configuration and the training is susceptible to peculiar lossy noises called ray-drop. To address the issue, this paper proposes a generative model of LiDAR range images applicable to the data-level domain transfer. Motivated by the fact that LiDAR measurement is based on point-by-point range imaging, we train an implicit image representation-based generative adversarial networks along with a differentiable ray-drop effect. We demonstrate the fidelity and diversity of our model in comparison with the point-based and image-based state-of-the-art generative models. We also showcase upsampling and restoration applications. Furthermore, we introduce a Sim2Real application for LiDAR semantic segmentation. We demonstrate that our method is effective as a realistic ray-drop simulator and outperforms state-of-the-art methods.

1. Introduction

A LiDAR sensor is a laser-based range sensor that can measure the surrounding geometry as a 3D point cloud. Compared to other depth cameras and radars, LiDAR sensors cover a wide field of view and are also robust to lighting conditions owing to their active sensing based on the pulsed laser. Therefore, LiDAR-based 3D perception has become an indispensable component of autonomous mobile robots and vehicles.

In particular, semantic segmentation on LiDAR point clouds [34, 35, 47, 48, 50, 53, 29] is one of the most important tasks in autonomous navigation, which identifies 3D objects at the point level for traversability estimation. For general point clouds, the most recent segmentation methods

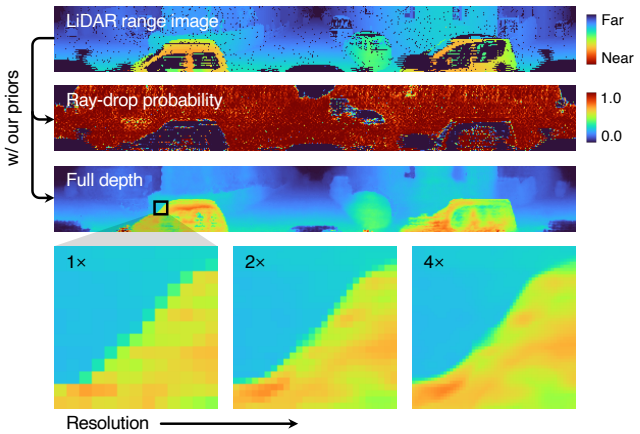


Figure 1. LiDAR range imaging (top) involves the missing points spreading unevenly, called ray-drops. Our method enables us to infer the ray-drop probability and the underlying complete scene (middle) on the continuous image representation (bottom).

are based on PointNet [34] and PointNet++ [35], neural network architectures designed to deal with the unordered nature of point clouds. However, they have limitations regarding computational speed [50] and memory requirements [3] for large-scale point clouds and are often performed on a reduced size. To further efficient training and inference, the spherical projection has been exploited in the LiDAR segmentation. In this approach, point clouds are represented as a bijective 2D grid, a so-called *range image* (see Fig. 1 for instance). So far, many studies [47, 48, 50, 53, 29] proposed 2D convolutional neural networks that perform point cloud segmentation on this range image representation.

While the range image representation has improved the processing efficiency, there are domain gap problems that degrade the performance in deploying trained models. In this paper, we argue two issues relating to ray-casting and ray-dropping. The ray-casting issue is derived from the angular configuration of emitted lasers. LiDAR sensors have a wide variety of angular resolutions due to hardware constraints, which might bring spatial bias in training the segmentation models. The ray-dropping issue is derived from

the phenomenon of laser reflection. While LiDAR sensors are robust to day-and-night illumination changes, produced range images involve quite a few missing points if the reflected laser intensity is too low due to scene-derived specular and diffuse reflection and light absorption. This artifact is problematic in simulation-to-real (Sim2Real) segmentation tasks because the phenomenon is non-trivial to be reproduced in a simulator. Some studies proposed LiDAR domain adaptation methods to address the ray-casting [23, 52] and ray-dropping issues [48, 47, 53, 27].

In this work, we propose a generative model-based method for LiDAR domain adaptation. Our method learns the generative process of LiDAR range images alongside the ray-casting and ray-dropping effects in an end-to-end manner, based on generative adversarial networks (GANs) [12]. The learned data priors can be used for mapping different domains. Our model builds upon the recently proposed two paradigms for generative models: implicit neural representation [41, 2] and lossy measurement model [6, 17, 30]. Implicit neural representation is a continuous and differentiable signal representation parameterized by neural networks. For example, an image is represented by a coordinate-based function and its resolution is determined by coordinate queries. Motivated by this scheme, we aim to model the editable ray-casting process of LiDAR range images. The lossy measurement model is an invertible function that simulates the stochastic signal corruption along the data generation. We aim to model the scene-dependent ray-dropping in an unsupervised manner.

In Section 4.1, we first evaluate our model in terms of generation fidelity and diversity compared to the point-based and image-based state-of-the-art generative models. Our model shows the best results in most standard image/point cloud metrics. We also examine the validity of feature-based metrics on LiDAR point clouds, motivated by de facto standard assessment in the natural image domain [14]. We then showcase applications with our model such as post-hoc upsampling and data restoration from sparse depth observation. Finally, in Section 4.2, we conduct Sim2Real semantic segmentation by using our model as a noise simulator. We demonstrate that our method produces realistic ray-drop noises and outperforms the state-of-the-art LiDAR Sim2Real methods. In summary, our contributions are as follows:

- We propose a novel GAN for LiDAR range images simulating the ray-casting and ray-dropping processes.
- We showcase the utility of our model on the post-hoc upsampling and data restoration.
- We apply our model to Sim2Real semantic segmentation. We empirically show that our model produces realistic ray-drop noises on simulation data and outperforms the state-of-the-art methods.

2. Related work

2.1. LiDAR domain adaptation

As in the field of natural images, the performance of LiDAR perception tasks also suffers from domain shift problems between the training and test environments [44] such as by different sensor configurations, geography, weather conditions, and simulation. We highlight the following two cases focused on in this work.

Angular resolution. Sampling angular resolution is a non-negligible property of LiDAR sensors, which determine the density of 3D point clouds. To mitigate the gap, Langer *et al.* [23] used pseudo LiDAR range images sampled from sequentially superimposed point clouds or meshes. However, the synthesis quality may depend on the sequential density of scans. Yi *et al.* [52] proposed voxel-based completion to bridge the scan-wise discrepancy, while this approach is designed for point-based perception methods. This paper proposes a scan-wise transfer using GAN-based data prior and shows some qualitative results.

Ray-drop noises. Ray-dropping occurs if the pulsed lasers failed to reflect from measured objects. This phenomenon is caused by complex physical factors such as mirror diffusion, specular diffusion, light absorption, and range limits. In the aspect of perception tasks, the ray-drop noises are one of the important properties of *real* data [47, 48, 53, 27]. Some studies tackled simulating the ray-drop noises to make the LiDAR simulator realistic. SqueezeSeg [47] proposed to sample binary noises based on the pixel-wise frequency computed from the real data. However, the averaged noises cannot make object-wise effects so they might be far from the actual distribution. To estimate ray-drop noises from LiDAR range images, Zhao *et al.* [53] trained CycleGAN [54] and Manivasagam *et al.* [27] trained U-Net [38]. However, those approaches cast the task as a binary classification based on a cross-entropy objective. As mentioned by Manivasagam *et al.* [27], the cross-entropy training does not guarantee the estimated probability is calibrated. We hypothesize that such approximated noise simulation might be suboptimal performance in Sim2Real tasks.

2.2. Deep generative modeling

In recent years, great progress has been achieved in generative models based on deep neural networks. In particular, a generative adversarial network (GAN) [12] has been attracting a great deal of attention in the image domain due to their sampling quality and efficiency [5]. As an example of recent studies, Karras *et al.* [18] proposed ProGAN for synthesizing mega-pixel natural images, and the generation quality has been significantly improved in a few years [20, 21, 19]. Moreover, well-trained GANs can be used as *generative image priors* for semantic manipulation and data restoration [13, 37].

Implicit neural representation. A GAN has also taken another step forward with implicit neural representation [2, 41]. Implicit neural representation is a method of continuous signal representation using a coordinate-based neural network. The typical architecture employs MLPs that receive arbitrary coordinate points and predict values such as signed distances [31] for 3D shapes, colors [2, 41] for 2D images, and colors/density [39] for volumetric rendering. This implicit scheme allows the models to learn the resolution-independent representation even if trained with discretized data such as images. CIPS [2] and INR-GAN [41] incorporated the idea into image GANs. They demonstrated that the models could control the resolution to perform spatial interpolation and extrapolation. This paper discusses the availability in modeling ray-casting.

Lossy measurement models. Training datasets are not always *clean* and can be problematic on training GANs. Since the objective of GANs is to mimic a distribution of a dataset, the learned generator space can also be noisy. To address the issue, some studies [6, 17, 30] introduced a *probabilistically* invertible function into the generative process to learn clean signals from only the noisy datasets. For instance on multiplicative binary noises, Bora *et al.* [6] proposed AmbientGAN and Li *et al.* [24] proposed MisGAN. Their noise models are formulated with a signal-independent probability, which does not meet the LiDAR's signal-dependent binary noises. Kaneko and Harada [17] proposed NR-GAN which can estimate the signal-dependent noise distribution; however binary noises are not covered.

LiDAR applications. Although most generative models have focused on the natural image datasets, several studies [7, 30] have begun to apply them to LiDAR data. Caccia *et al.* [7] proposed the first application of deep generative models on LiDAR data. They trained the variational autoencoders (VAEs) [22] on the range image representation. They also reported the visual results by a vanilla GAN [36]. However, the distribution of LiDAR range images can be discrete due to the random ray-drop noises, which is difficult to be represented by neural networks as a continuous function. Motivated by the concept of the lossy measurement models, Nakashima and Kurazume [30] proposed DUSty incorporating the differentiable ray-drop effect into GANs to robustly train the LiDAR range images. They employed straight-through Gumbel-Sigmoid distribution [26, 16] for modeling the LiDAR noises, so that the model learns the discrete data distribution as a composite of two modalities: an underlying complete depth and the corresponding uncertainty of measurement. Those two studies empirically showed the availability of generative models on LiDAR range images, while did not evaluate the effectiveness on the actual perception tasks. In contrast, our paper improves the generation quality and also provides the quantitative evaluation on Sim2Real semantic segmentation.

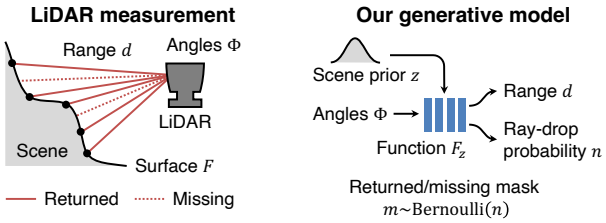


Figure 2. A schematic diagram of LiDAR measurement and our proposed generative model for 3D LiDAR data. We hypothesize that laser dropping occurs stochastically according to scene-specific probabilities.

3. Method

Fig. 2 depicts the LiDAR measurement and our formulation with deep generative models. Our aim is to learn the LiDAR scene priors independent of angular resolution and to leverage them for data-level domain transfer. To this end, Section 3.1 first introduces a resolution-free implicit representation of range images. In Section 3.2, we then present our GAN based on the implicit representation and differentiable ray-dropping effect. Finally, Section 3.3 introduces an inference step which uses our learned GAN as generative scene priors. We provide the implementation details in the supplementary materials. Our code is available at `https://github.com/kazuto1011/dusty-gan-v2`.

3.1. Implicit representation of range images

LiDAR range images. While a general representation of point clouds is a set of Cartesian coordinate points $\{(p_x, p_y, p_z)\}$ [1, 34, 51], LiDAR point clouds can also be represented as a bijective 2D grid [30, 7, 45, 53, 47, 48] due to the measurement mechanism, *range imaging*. Suppose that a LiDAR sensor emitting horizontal W pulsed lasers for H elevation angles measures a distance d for each angular position. Then all the distance values can be assigned to a $H \times W$ angular grid by spherical projection, where the resultant representation is called a range image. Each pixel has a set of sensor-dependent azimuth and elevation angles $\Phi = (\theta, \phi)$ and the corresponding distance d. Therefore, an arbitrary LiDAR scene $\{(p_x, p_y, p_z)\}$ can be seen as a set of spherical coordinates $\{(\theta, \phi, d)\}$ projected on the 2D grid.

Scenes as a function. The core idea of this paper is to represent the 3D scene by a function F mapping the spherical angular set to the distance: $d = F(\Phi)$. If the scene can be represented in the continuous function space F, we can reconstruct the scene with arbitrary resolution queries. Furthermore, it is possible to express multiple scenes by introducing parameters z that condition the function F. To this end, we aim to build the function $d = F(\Phi, z)$ as a deep generative model where the scene is instantiated by the sensor-agnostic scene prior z and queried by the sensor-specific angular set Φ to generate the LiDAR range images.

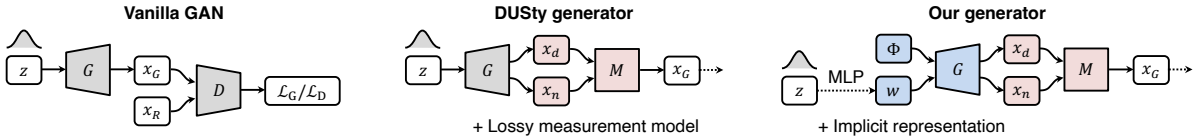


Figure 3. Comparison of the baseline GANs and ours. The vanilla GAN [7] consists of the generator G and the discriminator D and directly learns the distribution of raw range images x_R. DUSty generator [30] disentangles the generator space as the range x_d and measurability x_n (ray-drop probability) with the self-conditioned measurement model M. Our generator further introduces implicit neural representation [41] so that the spatial resolution is not fixed by the generator but controlled by the external query Φ.

3.2. Generative range imaging

Generative adversarial networks. To introduce the latent variable z that controls the scene, we build the function F as a generative model. We employ a generative adversarial network (GAN) [12], similar to prior work [30, 7]. A GAN typically consists of two networks: a generator G and a discriminator D. In image synthesis tasks, G maps a latent variable $z \sim N(0, I)$ to an image $x_G = G(z)$, whereas D tells the generated image x_G from sampled real images x_R. The networks are trained in an alternating fashion by minimizing the adversarial objective, *e.g.*, the following non-saturating loss [12]:

$$\mathcal{L}_D = -\mathbb{E}_x[\log D(x_R)] - \mathbb{E}_z[\log(1 - D(G(z)))], \quad (1)$$

$$\mathcal{L}_G = -\mathbb{E}_z[\log D(G(z))]. \quad (2)$$

In this paper, the generator G is equivalent to the aforementioned function F and the structure of x_G is represented by scene condition z and given coordinates Φ. Our GAN builds upon INR-GAN [41], which was demonstrated on natural images. INR-GAN first transforms the latent variable z to disentangled style space w by MLPs and modulates the network weights to synthesize images.

Lossy measurement model. LiDAR range images involve many missing points caused by ray-dropping. In the aspect of training GANs, the missing points prevent stability and fidelity of depth surfaces. To address this issue, we combine our model with the ray-dropping formulation proposed in DUSty [30]. They assumes the ray-dropping phenomenon is stochastic and uses Bernoulli sampling with self-conditioned probability. As outputs from the generator G, DUSty first assumes a complete range image $x_d \in \mathbb{R}^{H \times W}$ and the corresponding ray-drop probability map $x_n \in \mathbb{R}^{H \times W}$. Then, the final LiDAR measurement x_G is sampled from the complete x_d according to a lossy measurement mask $m \sim \text{Bernoulli}\,(x_n)$. Since the sampling m is non-differentiable, m is reparameterized with the straight-through Gumbel-Sigmoid distribution [16] to estimate the gradients. Note that the generator spaces x_d and x_n do not have to be discrete distribution, while x_G can produce discrete noises caused by ray-dropping. As in DUSty, we convert each distance value x_d into the *inverse* depth for further stable training. In Fig. 3, we compare the vanilla GAN [7], DUSty [30], and our proposed model.

Positional encoding for circular grid. In the fields of implicit neural representation [41, 2], positional encoding is an indispensable technique to represent high-frequency details of output images, which transforms coordinates points to high-dimensional feature space. In particular, Fourier features [43] are the most popular encoding scheme in the image domain where the coordinate $\Phi = (\theta, \phi)$ is transformed by the following sinusoidal function:

$$\text{PE}(\theta, \phi) = \sin([b_\theta, b_\phi][\theta, \phi]^\top), \quad (3)$$

where $b_\theta \in \mathbb{R}^D$ and $b_\phi \in \mathbb{R}^D$ are weight vectors which control frequencies in encoded space, and $\theta, \phi \in [-\pi, \pi]$ are angular values determined by LiDARs. In natural image applications, the weights can be set by various formula such as a power of two [28], Gaussian samples [43], and learnable parameters [2, 41]. In our case, the weights should be carefully initialized so that the encoding preserves the cylindrical structure of the angular input. First, we set the limit value of output frequencies both for azimuth and elevation inputs. We then uniformly sample b_ϕ within the determined limit and sample b_θ from a set of power of two.

Subgrid training. The sampled frequencies in positional encoding are sparse in the horizontal direction. If the subsequent blocks are overfitted with a small number of fixed sampling points, post-hoc changes or upsampling in the angular input may result in unexpected aliasing at intermediate layers. This is not compatible with our purpose to control the generation format according to the sensor specification. To this end, we augment the angular inputs by a random phase shift horizontally during training and translate the output images in inverse direction for the number of corresponding pixels.

3.3. Domain-agnostic inference

This section describes a method to reconstruct and compensate for the arbitrary LiDAR measurement using *GAN inversion* [49]. The GAN inversion is an inference task to find the latent representation of the given data. The standard approaches for GAN inversion have two camps: auto-decoding [37, 21] which optimizes the latent codes to match the data, and training additional encoders [33] to estimate the latent code directly. In this paper, we employ Pivotal Tuning Inversion (PTI) [37], one of the latest auto-decoding

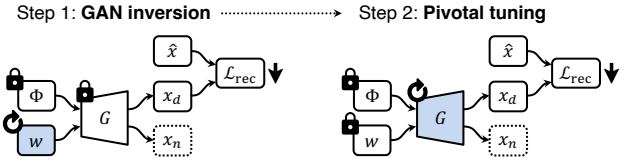


Figure 4. Overview of our inference method based on PTI [37]. Step 1 optimizes the latent code w and step 2 fine-tunes weights Ω of the generator G. The by-product x_n can be used for Sim2Real applications.

approaches. This section briefly introduce its steps along with our designed optimization objective. Fig. 4 shows the overview of our method.

Step 1: GAN inversion. As the latent representation, we use the aforementioned style code w instead of z. Let $\hat{x}$ and $\hat{m}$ are a target depth map and the corresponding ray-drop mask. We first define the following objective to assess the depth error:

$$\mathcal{L}_{\text{rec}} = \frac{\|\hat{m} \odot (1 - x_d(w, \Phi; \Omega)/\hat{x})\|_1}{\|\hat{m}\|_1}, \tag{4}$$

where $x_d(w, \Phi; \Omega)$ is a generated depth map conditioned by the latent code w, resolution query Φ, and generator weights Ω. The objective $\mathcal{L}_{\text{rec}}$ measures the relative absolute error normalized by the valid points in $\hat{m}$. In this step, we compute the matching code by $\hat{w} = \arg\min_w \mathcal{L}_{\text{rec}}$.

Step 2: pivotal tuning. With the optimized $\hat{w}$, we can generate a range image x_G similar to the target $\hat{x}$. This step further minimize the minor appearance difference by fine-tuning the generator weights Ω while freezing the pre-optimized code $\hat{w}$ to pivot on the fundamental structure. We minimize the same objective with respect to Ω: $\hat{\Omega} = \arg\min_\Omega \mathcal{L}_{\text{rec}}$.

Since our masked objective $\mathcal{L}_{\text{rec}}$ relies on only measurable points, we can also apply the inference to restore partially observed images. Moreover, we can obtain the by-product x_n by reconstructing simulation data as $\hat{x}$, which can be used to simulate the ray-drop noises.

4. Evaluation

4.1. Generation fidelity and diversity

Following the related studies on generative models, we first evaluate fidelity and diversity of generated samples.

Dataset. We use the KITTI Raw dataset [11] since the other popular releases such as KITTI Odometry [11] include missing artifacts by ego-motion correction [45]. The dataset provides 22 trajectories of scans measured by the Velodyne HDL-64E LiDAR, where each scan has 64 layers vertically. For future derivative work, we use standard splits defined by KITTI Odometry [11][1]. The provided data are sequences of Cartesian coordinate points ordered by angles. We first chunk the ordered sequence into 64 sub-sequences, where each represents one elevation angle. We then subsample 512 points for each sub-sequence and stack them to form a 64×512 range image.

Baselines. We compare our model with two popular baselines of point-based generative models: r-GAN [1] and l-WGAN [1]. r-GAN is a kind of GANs based on point set representation, which consists of an MLP generator and a PointNet [34] discriminator. l-WGAN first learns a PointNet-MLP autoencoder and then performs adversarial training with the additional MLPs to generate the bottleneck feature. We train l-WGAN autoencoder by measuring the earth mover's distance (EMD) between input and reconstruted point clouds. The size of LiDAR point clouds is much larger than the typical benchmark for point-based methods and the EMD distance calculation takes extremely high computational cost both in training and evaluation. For efficiency and fareness, we first downsample the LiDAR point clouds to the conventional number of points 2048 by farthest point sampling (FPS). We also compare with two types of image-based generative models: a vanilla GAN [7] and DUSty [30]. The vanilla GAN and DUSty share the backbone design based on 4×4 transposed convolution, while DUSty is adopted the ray-drop measurement model explained in Section 3.2. Those models cannot change the output resolution from the training configuration.

Point-based metrics. Following the related work on the point-based generative models [1, 51, 30], we measured four types of distributional similarities between the sets of reference and generated point clouds, defined by Yang *et al.* [51]: Jensen–Shannon divergence (JSD) for fidelity, coverage (COV) for diversity, minimum matching distance (MMD) for fidelity, and 1-nearest neighbor accuracy (1-NNA) for both fidelity and diversity evaluation. To compute the distance between point clouds for COV, MMD, and 1-NNA, we used EMD.

Image-based metrics. Additionally, we computed the sliced Wasserstein distance (SWD) [18] to measure the patch-based image similarity for evaluating quality of the inverse depth maps. SWD is computed based on 7×7 patches extracted from three level of image pyramids. For all metrics, we report the mean scores over three runs with different seeds.

Feature-based metrics. Existing evaluation metrics [51] for synthesized point clouds is impracticable for large number of points due to the limited scalability of sample-to-sample distance computation such as EMD. Nakashima *et*

[1] We use 18,329 scans for training (sequence 3 is not available), 4,071 scans for validation from "city", "road", and "residential" categories. We define a test set by the remaining 18,755 scans since the correspondence is not publicly available.

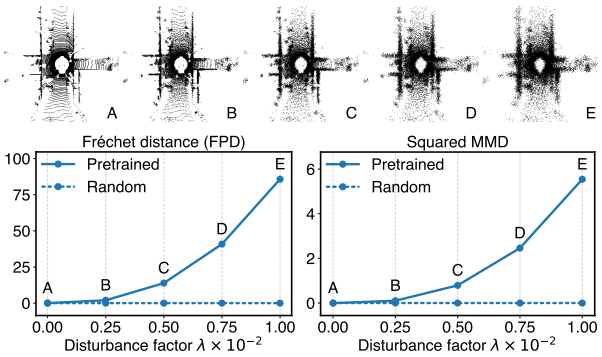


Figure 5. Sanity checks of the Fréchet distance (FPD [40]) and the squared MMD [4] on feature representation of LiDAR point clouds. We applied additive Gaussian noises with a coefficient λ to the KITTI point clouds (see A–E) and computed the metrics with the *clean* original point clouds. All point clouds were encoded by PointNet [34] with pre-trained or random weights.

al. [30] addressed the issue on LiDAR point clouds by subsampling real and generated point clouds in evaluation. As an alternative metrics, Shu *et al.* [40] proposed Fréchet point cloud distance (FPD) as an analogy of FID [14] which is the de facto standard metrics in the image domain. FPD first maps an arbitrary number of points to a low-dimensional feature space by PointNet pre-trained on ShapeNet [8], then measures the Fréchet distance between the real and generated data distributions in the feature space. Unlike the aforementioned point cloud metrics, even large-scale point clouds such as LiDAR data can be calculated without downsampling. Following that Heusel *et al.* [14] evaluated the sensitivity of FID under various image disturbances, we first confirm the validity of the off-the-shelf PointNet on LiDAR point clouds. We apply additive Gaussian noises with a coefficient λ to the KITTI training set and compute FPD against the clean original training set. On PointNet features, we also compute squared maximum mean discrepancy (squared MMD) [4] which is also used in the image domain as KID. Fig. 5 shows that as the strength of the perturbation increases, both metrics reflect the dissimilarity. The further results under various disturbance effects are provided in our supplementary material.

Quantitative comparison. Table 1 reports FPD and squared MMD. We can see that our method outperformed both the point-based and image-based baselines with large margins. Table 2 reports SWD for image-based methods. Our method showed the best performance even in this 2D level assessment. Finally, Table 3 reports JSD, COV, MMD, and 1-NNA. Except for JSD and MMD, the proposed method showed the highest score. As for l-WGAN showing the best in MMD, we believe this is due to the fact that it optimizes EMD directly in training of the autoencoder. We provide a generated samples of all methods in the supplementary materials.

Table 1. Quantitative comparison by distributional similarity of PointNet features: FPD and MMD^2.

	2048 points		64×512 (full)	
Method	FPD ↓	MMD^2 ↓	FPD ↓	MMD^2 ↓
r-GAN [1]	787.45	45.02	–	–
l-WGAN [1]	129.35	10.65	–	–
Vanilla GAN [7]	3629.36	671.14	3648.68	675.24
DUSty [30]	232.90	39.62	241.32	42.66
Ours	**96.11**	**3.66**	**93.85**	**3.84**

Table 2. Quantitative comparison by distributional similarity of inverse depth maps: sliced Wasserstein distance (SWD ↓).

Method	16×128	32×256	64×512	Mean
Vanilla GAN [7]	0.397	0.371	0.746	0.505
DUSty [30]	**0.353**	0.353	0.768	0.491
Ours	0.378	**0.278**	**0.611**	**0.422**
Training set	0.257	0.207	0.765	0.410

Table 3. Quantitative comparison by distributional similarity of point clouds: JSD$\times 10^2$, COV, MMD$\times 10^2$, and 1-NNA.

Method	JSD ↓	COV ↑	MMD ↓	1-NNA ↓
r-GAN [1]	21.73	0.013	17.51	1.000
l-WGAN [1]	4.91	0.324	**8.62**	0.896
Vanilla GAN [7]	10.31	0.290	12.34	0.986
DUSty [30]	**3.00**	0.375	9.41	0.898
Ours	3.04	**0.388**	9.12	**0.892**
Training set	2.80	0.362	0.765	0.890

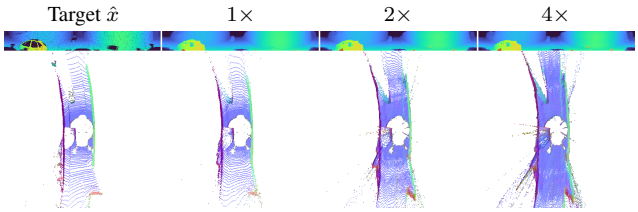


Figure 6. LiDAR data upsampling. From left to right, the target range image $\hat{x}$ and our reconstruction results x_d in $1\times$, $2\times$, and $4\times$ resolutions. The bottom row shows bird's-eye views of the corresponding point clouds.

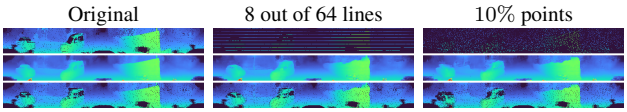


Figure 7. LiDAR data restoration. From top to bottom, the target range images $\hat{x}$, our reconstruction results x_d, and the ones with rendered ray-drops x_G.

Applications. Figs. 6 and 7 show upsampling and restoration results, respectively. Both are obtained by our auto-decoding method introduced in Section 3.3. From the reasonable results, we consider our model successfully learned the scene priors of LiDAR range images.

Table 4. Quantitative comparison of Sim2Real semantic segmentation results. We train SqueezeSegV2 [48] on GTA-LIDAR [48] (simulation domain) and then evaluate precision (%, ↑), recall (%, ↑), and IoU (%, ↑) on KITTI-frontal [48] (real domain).

			Car			Pedestrian			
Config	Training domain	+ Ray-drop prior	Precision	Recall	IoU	Precision	Recall	IoU	mIoU
A	Simulation		54.2	1.1	1.1	27.7	2.5	2.4	1.7
B	Simulation	+ Global frequency	66.2	77.0	55.2	29.9	61.0	25.1	40.2
C	Simulation	+ Pixel-wise frequency [48]	72.9	75.5	59.0	26.1	62.0	22.5	40.7
D	Simulation	+ Auto-decoding w/ DUSty [30]	72.0	76.8	59.1	**34.5**	59.6	**28.0**	43.5
E	Simulation	+ **Auto-decoding w/ ours**	**74.8**	**87.0**	**67.3**	28.8	**67.1**	25.2	**46.3**
F	Real		78.7	86.5	70.1	66.5	18.0	16.5	43.3

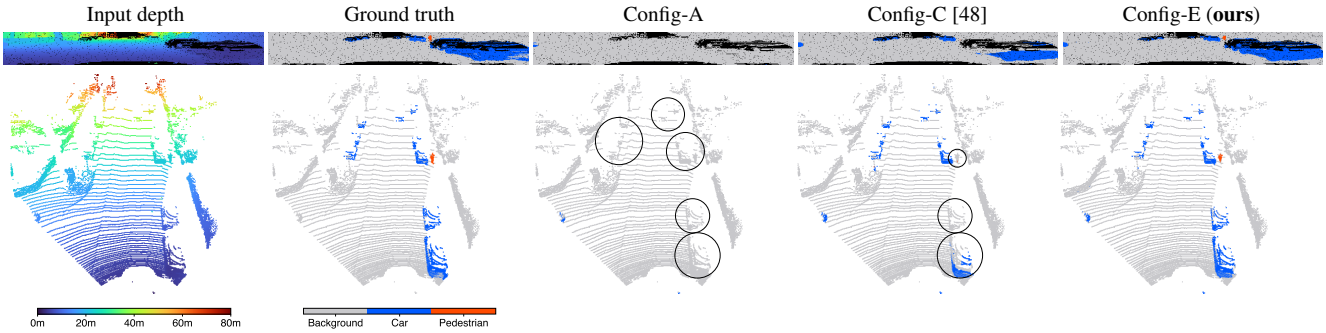


Figure 8. Qualitative comparison of Sim2Real semantic segmentation results. We can see that ours (config E) reduced the false negative on the *car* regions, as supported by the recall improvement in Table 4.

4.2. Sim2Real semantic segmentation

Model-based LiDAR simulators [47, 10] can produce a large amount of annotated training data, while there is an appearance gap against the real domain since the ray-drop noises are ignored or approximated. Some studies address the issue by leveraging the ray-drop frequency [47] or learning inference networks [53, 27]. As addressed in Section 3.3, our auto-decoding process can also generate a ray-drop probability map x_n by reconstructing valid points in a given range image $\hat{x}$. This motivates us to reproduce the pseudo ray-drop noises on the simulated range images. In this sections, we demonstrate the effectiveness of our method on the Sim2Real semantic segmentation.

Datasets. We follow the experiment protocol by Wu *et al.* [48] where a segmentation model is trained on the GTA-LiDAR dataset [48] and evaluated on the 90° frontal subset [48] of the KITTI dataset [11], hereinafter called KITTI-frontal. GTA-LiDAR is composed of 120k in-game LiDAR range images annotated with pixel-wise labels for *car* and *pedestrian* classes. KITTI-frontal is composed of 10k real range images subsampled from KITTI, which is also annotated for the same classes. KITTI-frontal contains 8,057 images for training and 2,791 images for testing.

Our approach. Before training the segmentation model, we perform the auto-decoding on each sample of GTA-LiDAR and obtain the corresponding ray-drop probability map x_n. At training phase, we sample Bernoulli noises from x_n and render the ray-drop noises on the fly.

Baselines. Our experiments are composed of two parts. The first experiment compares five approaches with different ray-drop priors, as listed in Table 4. Config-A is a baseline without rendering ray-drop noises. In config-B, we sample Bernoulli noises from global frequency computed from all pixels in KITTI-frontal. Config-C is the approach used by Wu *et al.* [48], where the noises are sampled from pixel-wise frequency of KITTI-frontal. Finally, config-D and config-E are the GAN-based auto-decoding approach. Config-D uses DUSty [30], while config-E uses our proposed GAN. Both models are pretrained on KITTI in Section 4.1. For comparison, we also provide the oracle results trained on KITTI-frontal (config-F). We use SqueezeSegV2 [48] for the architecture of semantic segmentation. To demonstrate the exclusive effect of noise rendering, we do not use any other adaptation techniques used in SqueezeSegV2, such as learned intensity rendering, geodesic correlation alignment, and progressive domain calibration. In the second experiment, we compare our model (config-E) with state-of-the-art domain adaptation methods: DAN [25], CORAL [42], HoMM [9], ADDA [46], CyCADA [15], and ePointDA [53]. ePointDA is a CycleGAN-based method and closely related to us in that simulating ray-drop noises on range images.

Results. Table 4 reports intersection-over-union (IoU) for each class and the mean score (mIoU). Fig. 8 provides visual comparison of config-A without noises, config-C [48], and our config-E. Although SqueezeSegV2 is already de-

Table 5. Sim2Real performance in comparison with state-of-the-art domain adaptation (DA) methods. Following the prior work, we report precision (%, ↑), recall (%, ↑), and IoU (%, ↑) for each class. The scores of DAN [25], CORAL [42], HoMM [9], ADDA [46], and CyCADA [15] are from the report by Zhao *et al.* [53].

	DA†		Input modality‡				Car			Pedestrian			
Method	D	F	C	R	I	M	Precision	Recall	IoU	Precision	Recall	IoU	mIoU
SqueezeSegV2 [48]§	✓	✓	✓	✓	✓	✓	–	–	57.4	–	–	23.5	40.5
DAN [25]		✓	✓				56.3	76.4	47.8	20.8	**68.9**	19.0	33.4
CORAL [42]		✓	✓				56.5	82.1	50.2	26.0	50.3	20.7	35.5
HoMM [9]		✓	✓				59.4	85.2	53.9	26.2	66.8	23.2	38.6
ADDA [46]		✓	✓				56.7	83.5	50.7	24.7	58.5	21.0	35.9
CyCADA [15]	✓	✓	✓				40.9	72.1	35.3	17.8	52.4	15.3	25.3
ePointDA [53]§	✓	✓	✓				73.4	81.9	63.4	**29.4**	56.0	23.9	43.7
ePointDA [53]	✓	✓	✓				**75.2**	84.7	66.2	28.7	65.2	24.8	45.5
Ours (config-E)§	✓		✓	✓			74.8	**87.0**	**67.3**	28.8	67.1	**25.2**	**46.3**

† Category of domain adaptation (DA): D: data-level DA and/or F: feature-level DA.
‡ C: the Cartesian coordinates, R: depth, I: estimated intensity [48], M: a binary mask indicating either measured or missing points.
§ The same model architecture (SqueezeSegV2 [48]), but the different DA approach and input modality.

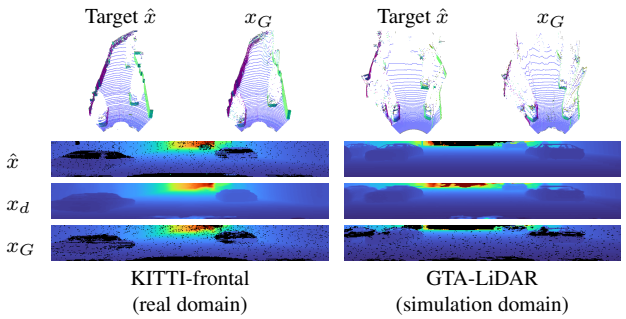


Figure 9. Auto-decoding examples by our method. We show the targets $\hat{x}$, the intermediate outputs x_d, and the final outputs x_G.

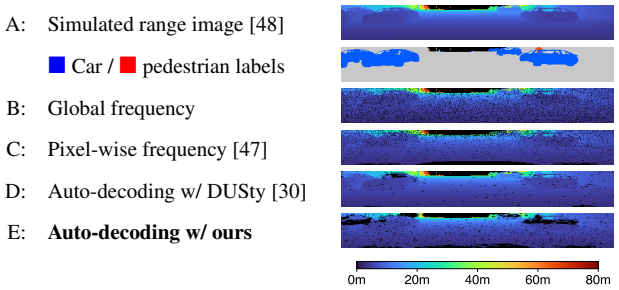


Figure 10. Qualitative comparison of noise rendering methods on GTA-LiDAR [48].

signed to reduce the ray-drop sensitivity, the performance is extremely low without noise rendering (config-A). Surprisingly, even simple rendering with the global frequency (config-B) boosted all the metrics and its spatial extension by config-C [47] brought subtle improvement. GAN-based approaches (config-D and config-E) further improved the results. In particular, our model (config-E) showed the best mIoU and surpassed the results on real domain (config-F). Fig. 9 shows the auto-decoded examples and Fig. 10 compares rendered noises. We can see that our method successfully simulated instance-level ray-drops such as on the car body and ego-vehicle shadows. In contrast, the results by the global frequency and the pixel-wise frequency are approximated. Finally, Table 5 compares our results (config-E) with state-of-the-art domain adaptation methods. Despite not applying any domain adaptation techniques other than rendering noises, our model showed the best IoUs.

5. Conclusion

In this paper, we introduced a novel approach for learning data priors of 3D LiDAR data towards domain adaptation applications. Our key idea is to represent LiDAR range images by a coordinate-based generative model and to learn clean data space through a pseudo-measurement model. We designed our model based on state-of-the-art GANs and demonstrated its effectiveness in the LiDAR domain. First, we evaluated the generation fidelity and diversity of sampled data. Our model showed superior results against the image-based and point-based baselines. We also conducted a Sim2Real semantic segmentation using our learned ray-drop priors. Our instance-level noise simulation brought significant improvement qualitatively and quantitatively and outperformed state-of-the-art methods. The results revealed that rendering ray-drop noises is important to mitigate a gap between the real and simulation domains. We consider our sensor-agnostic scene representation has the potential for cross-dataset tasks. Future work includes domain adaptation between different LiDARs and mixing accessible datasets for further training.

Acknowledgements

This work was partially supported by a Grant-in-Aid for JSPS Fellows Grant Number JP19J12159, JSPS KAKENHI Grant Number JP20H00230, and JST [Moonshot R&D] [Grant Number JPMJMS2032].

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] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

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

[5] Sam Bond-Taylor, Adam Leach, Yang Long, and Chris G. Willcocks. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 44(11):7327–7347, 2022.

[6] Ashish Bora, Eric Price, and Alexandros G Dimakis. AmbientGAN: Generative models from lossy measurements. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[7] 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.

[8] 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.

[9] Chao Chen, Zhihang Fu, Zhihong Chen, Sheng Jin, Zhaowei Cheng, Xinyu Jin, and Xian-Sheng Hua. HoMM: Higher-order moment matching for unsupervised domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3422–3429, 2020.

[10] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the Annual Conference on Robot Learning (CoRL)*, pages 1–16, 2017.

[11] 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.

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 2672–2680, 2014.

[13] Jinjin Gu, Yujun Shen, and Bolei Zhou. Image processing using multi-code gan prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3012–3021, 2020.

[14] 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.

[15] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. CyCADA: Cycle-consistent adversarial domain adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1989–1998, 2018.

[16] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[17] 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.

[18] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[19] 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.

[20] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4401–4410, 2019.

[21] 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.

[22] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

[23] Ferdinand Langer, Andres Milioto, Alexandre Haag, Jens Behley, and Cyrill Stachniss. Domain transfer for semantic segmentation of lidar data using deep neural networks. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8263–8270, 2020.

[24] Steven Cheng-Xian Li, Bo Jiang, and Benjamin Marlin. MisGAN: Learning from incomplete data with generative adversarial networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[25] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 97–105, 2015.

[26] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[27] Sivabalan Manivasagam, Shenlong Wang, Kelvin Wong, Wenyuan Zeng, Mikita Sazanovich, Shuhan Tan, Bin Yang, Wei-Chiu Ma, and Raquel Urtasun. LiDARsim: Realistic lidar simulation by leveraging the real world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11167–11176, 2020.

[28] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 405–421, 2020.

[29] Andres Milioto, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. RangeNet++: Fast and accurate lidar semantic segmentation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4213–4220, 2019.

[30] 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.

[31] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 165–174, 2019.

[32] 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.

[33] Guim Perarnau, Joost van de Weijer, Bogdan Raducanu, and Jose M. Álvarez. Invertible conditional gans for image editing. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS) workshop*, 2016.

[34] 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.

[35] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

[36] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[37] 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.

[38] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241, 2015.

[39] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. GRAF: Generative radiance fields for 3d-aware image synthesis. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 20154–20166, 2020.

[40] 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.

[41] 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.

[42] Baochen Sun and Kate Saenko. Deep CORAL: Correlation alignment for deep domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 443–450, 2016.

[43] 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.

[44] Larissa T Triess, Mariella Dreissig, Christoph B Rist, and J Marius Zöllner. A survey on deep domain adaptation for lidar perception. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV) Workshops*, pages 350–357, 2021.

[45] Larissa T. Triess, David Peter, Christoph B. Rist, and J. Marius Zöllner. Scan-based semantic segmentation of lidar point clouds: An experimental study. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, 2020.

[46] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7167–7176, 2017.

[47] 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.

[48] Bichen Wu, Xuanyu Zhou, Sicheng Zhao, Xiangyu Yue, and Kurt Keutzer. SqueezeSegV2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4376–4382, 2019.

[49] Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. Gan inversion: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2022.

[50] Chenfeng Xu, Bichen Wu, Zining Wang, Wei Zhan, Peter Vajda, Kurt Keutzer, and Masayoshi Tomizuka. SqueezeSegV3: Spatially-adaptive convolution for efficient point-cloud segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 1–19, 2020.

[51] 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.

[52] Li Yi, Boqing Gong, and Thomas Funkhouser. Complete & label: A domain adaptation approach to semantic segmentation of lidar point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15363–15373, 2021.

[53] Sicheng Zhao, Yezhen Wang, Bo Li, Bichen Wu, Yang Gao, Pengfei Xu, Trevor Darrell, and Kurt Keutzer. ePointDA: An end-to-end simulation-to-real domain adaptation framework for lidar point cloud segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3500–3509, 2021.

[54] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2223–2232, 2017.

Supplementary Material

A. Overview

This supplementary material summarizes implementation details of our model architectures and experiments in Section B, detailed analysis of evaluation metrics in Section C, generated examples of our method and baselines in Section D, and Sim2Real semantic segmentation results in Section E.

B. Implementation details

B.1. Models

Fig. 11 shows an overview of our proposed GAN framework. We design the generator network based on INR-GAN [41], 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. 11a. 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 [43]. 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. 11c, we use the same setup of DUSty [30] while replace the backbone with StyleGAN2 [21]. We applied the separable blur filter [17] to the discriminator inputs and modify all the kernels with circular padding.

B.2. Training

We employed the adaptive discriminator augmentation (ADA) [19] for all the image-based methods: vanilla GAN, DUSty, and ours. The augmentation basically followed the original pipeline by Karras *et al.* [19], 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 [21]. 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.

B.3. Computational cost of EMD

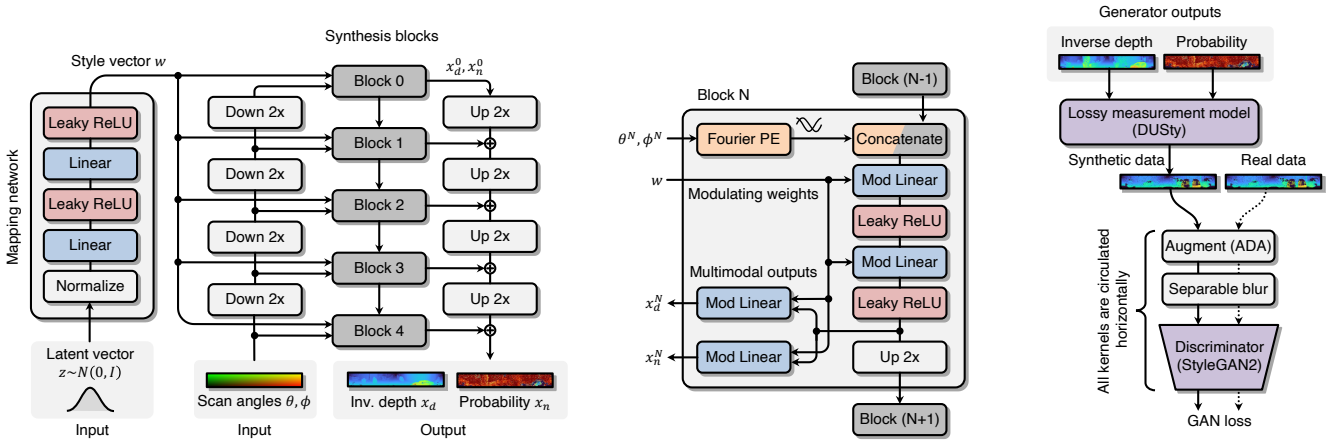
Earth mover's distance (EMD) is one of the metrics measuring the error between point clouds. Compared to the other metrics such as chamfer distance, EMD reflects the local details and the density distribution and is popular for the assessment of point clouds. However, it is known that computing EMD has an $o(N^3)$ complexity where N is the number of points in 3D point clouds [32]. 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 [51]. In Fig. 12, 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.* [30], 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.

B.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 [21, 41, 2, 19, 37]. 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.

C. Sanity check of evaluation metrics

For evaluating GANs, we used two types of distributional metrics on the PointNet representation: Fréchet distance [40] (named FPD for point clouds), squared maximum mean discrepancy (squared MMD) [4]. 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 [40] has been proposed for evaluating ShepeNet [8] generation task where each sample forms small-scale point clouds uniformly sampled from CAD objects. Squared MMD [4] was used to extend Fréchet Inception distance (FID) [14] 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 [34] provided by Shu *et al.* [40]. The Point-



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

Figure 11. Building blocks of our proposed GAN framework.

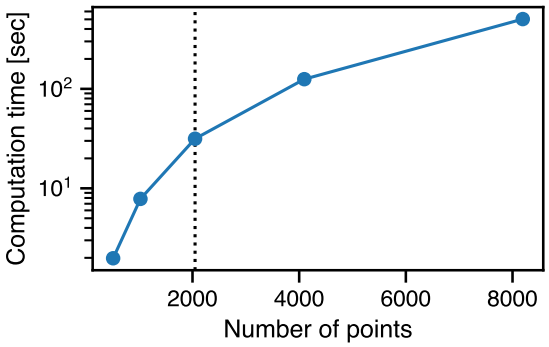


Figure 12. EMD computation time as a function of the number of points. The conventional number of point cloud tasks is 2048 (dotted line), while our task uses $64 \times 512 = 32,768$ points in full setting.

Net backbone[2] is pre-trained on the ShapeNet dataset and used by the original FPD [40]. 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. 13, 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. 13c–f. Although there are scale gaps depending on the type of disturbance, the results are roughly similar to the sanity check of FID [14]. Therefore, we concluded that the two metrics can be used to evaluate the generative models on LiDAR point clouds.

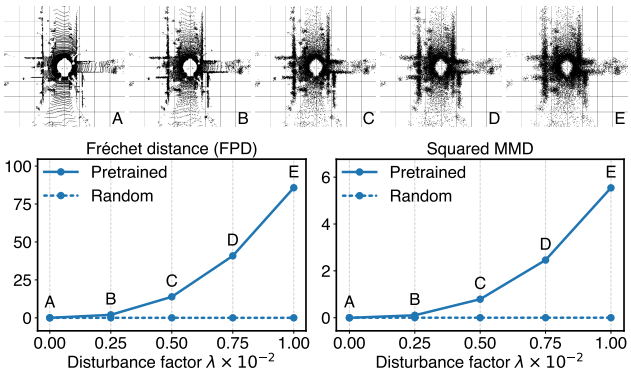
[2] https://github.com/seowok/TreeGAN

D. Generated examples

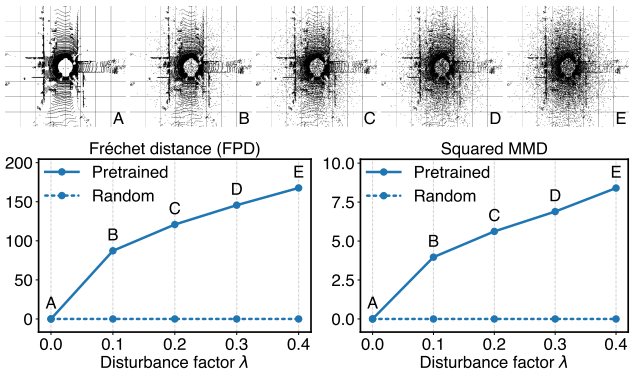
In Fig. 14, we provide uncurated sets of real and generated samples from image-based methods including ours. Fig. 15 compares the results between ours and the most closely related work, DUSty [30]. A close-up comparison shows that baseline methods include checkerboard artifacts and our method succeeded in expressing the smooth road surface. In Fig. 16, 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. 17, 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.

E. Sim2Real semantic segmentation

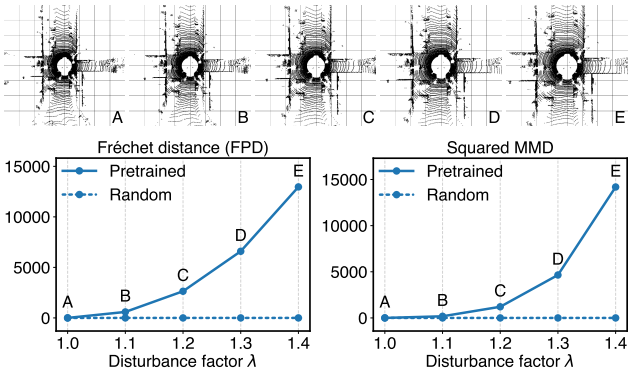
In Fig. 18, we show Sim2Real segmentation results on KITTI-frontal [48]. All models are trained on GTA-LiDAR while the ray-drop priors are different. We can see that our method (config-E) greatly improved the false negative regions of car classes.



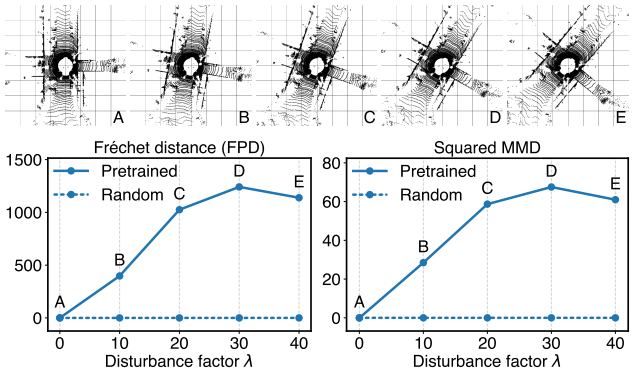
(a) Additive Gaussian noises with a coefficient λ



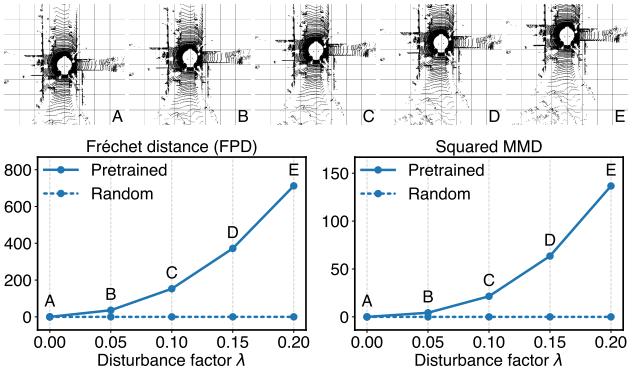
(b) Drop-in Gaussian noises for $\lambda \times 100$ (%) of points



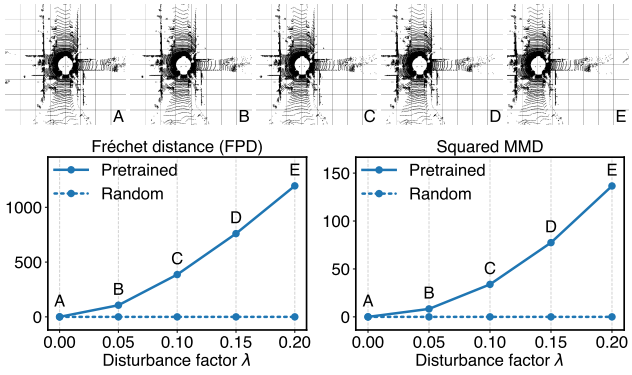
(c) Inflating coordinates with a multiplicative factor λ



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

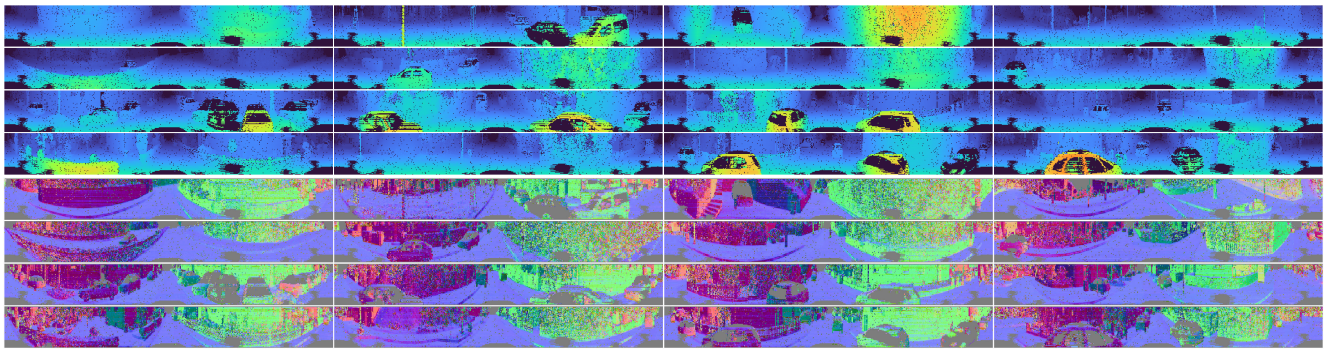


(e) Translation in x direction by λ

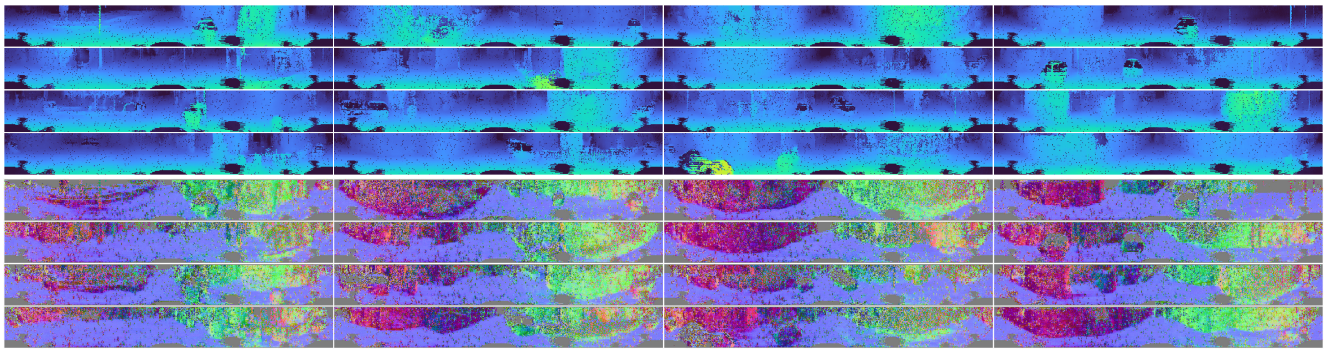


(f) Translation in y direction by λ

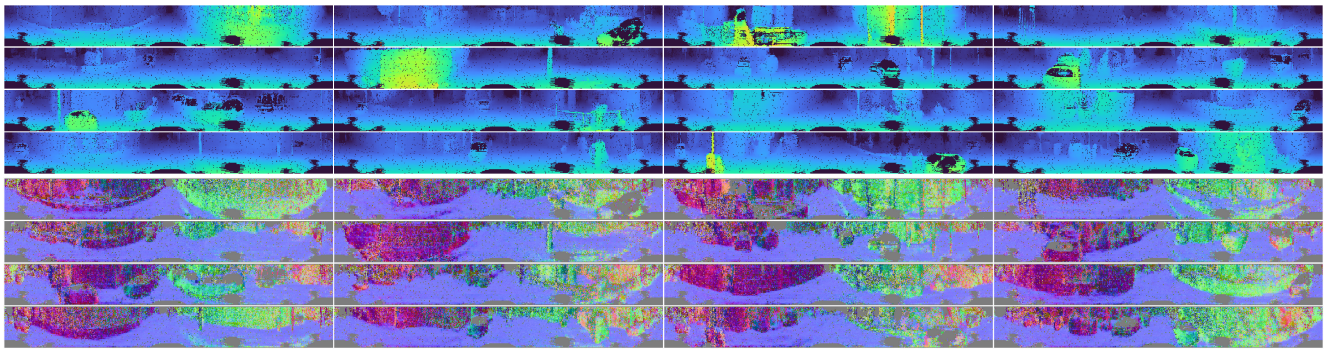
Figure 13. Disturbance sensitivity of four metrics: FPD [40] (Fréchet distance for point clouds) and squared maximum mean discrepancy (squared MMD) [4]. We applied six types of disturbances to the KITTI point clouds with various strength (see A–E) and computed the metrics with the *clean* original point clouds. All point clouds were encoded by PointNet [34] with pre-trained or random weights.



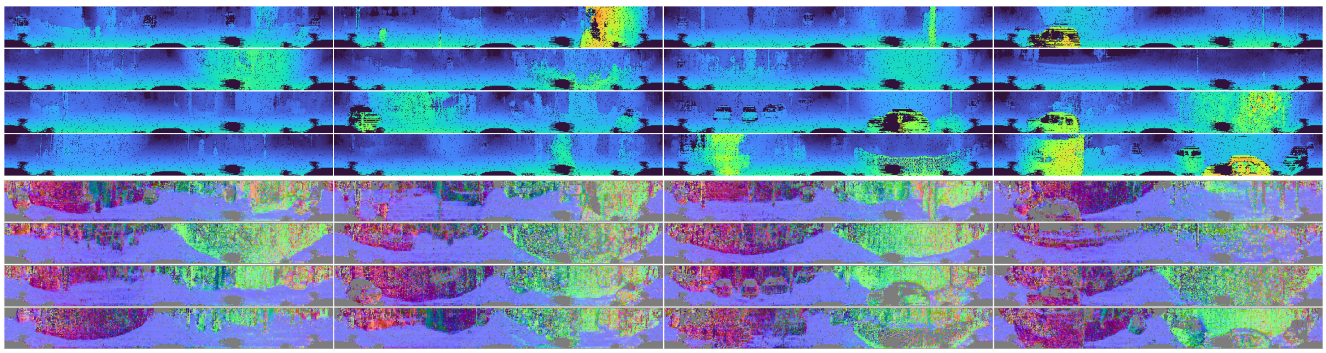
Real data



Vanilla GAN [7]



DUSty [30]



Ours

Figure 14. 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.

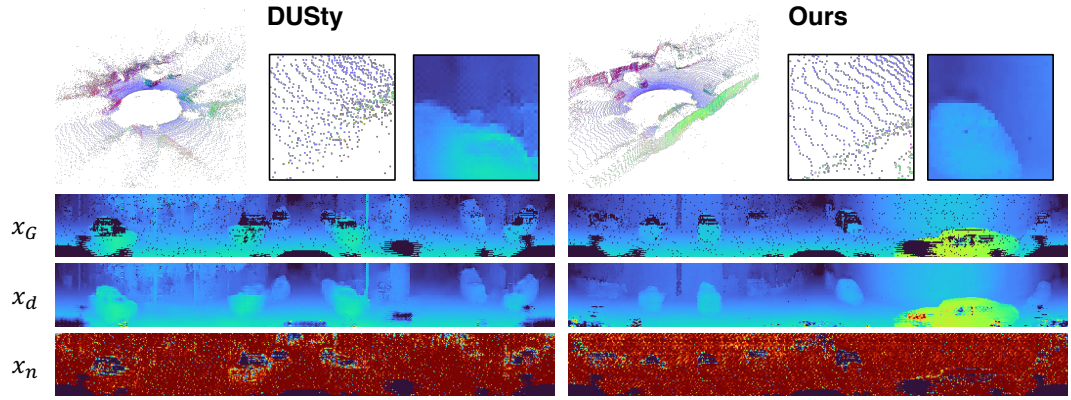


Figure 15. Qualitative comparison between DUSty [30] and ours. From top to bottom: generated point clouds, the final inverse depth maps x_G, the complete depth maps x_d, and the ray-drop probability maps x_n.

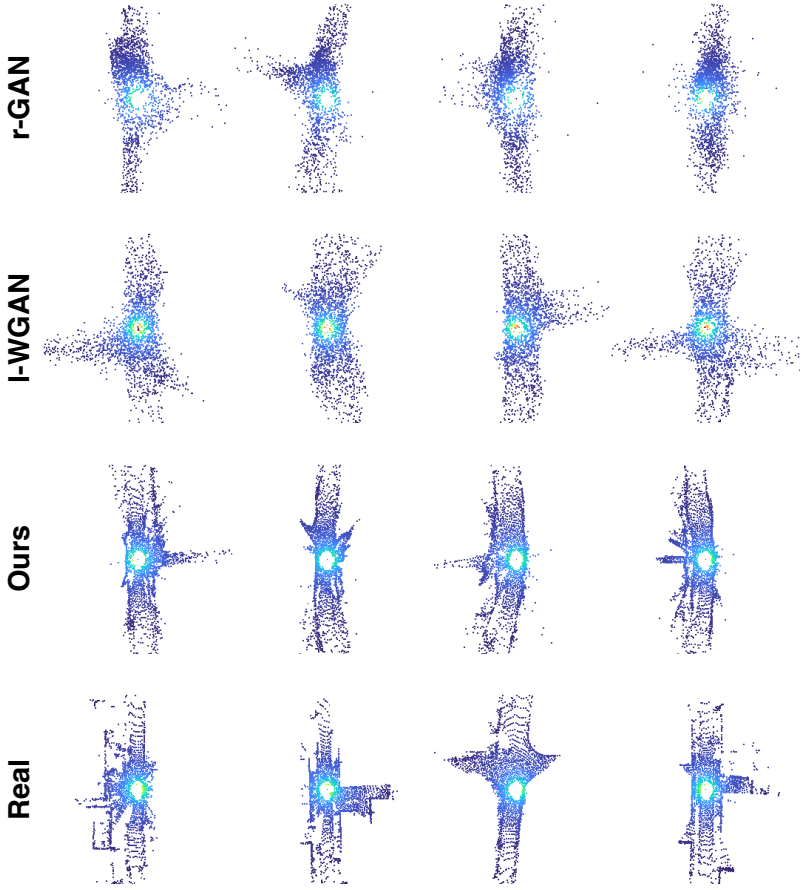


Figure 16. Qualitative comparison in bird's eye views of real and generated point clouds.

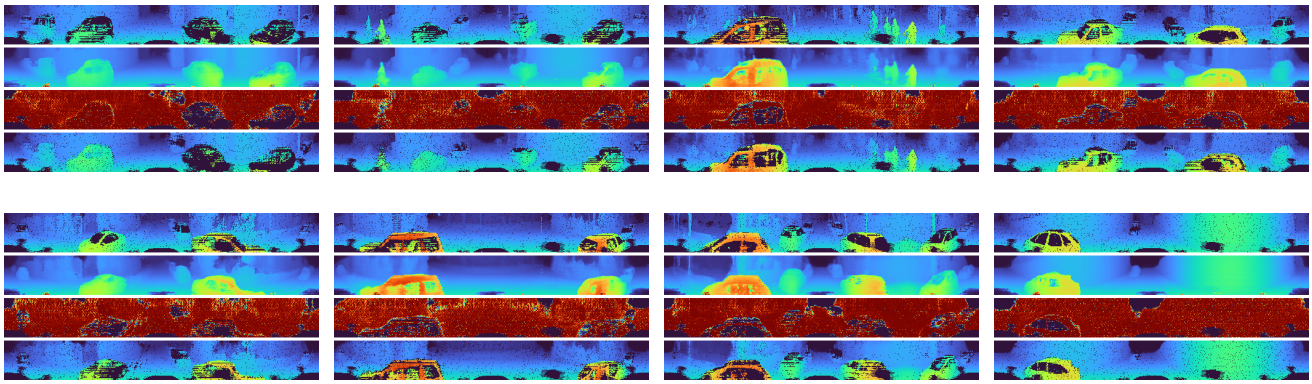


Figure 17. Reconstruction examples by our auto-decoding method. For each group, from top to bottom, we show the target range image $\hat{x}$ from KITTI [11], the generated inverse depth map x_d, the generated ray-drop probability map x_n, and the final output x_G.

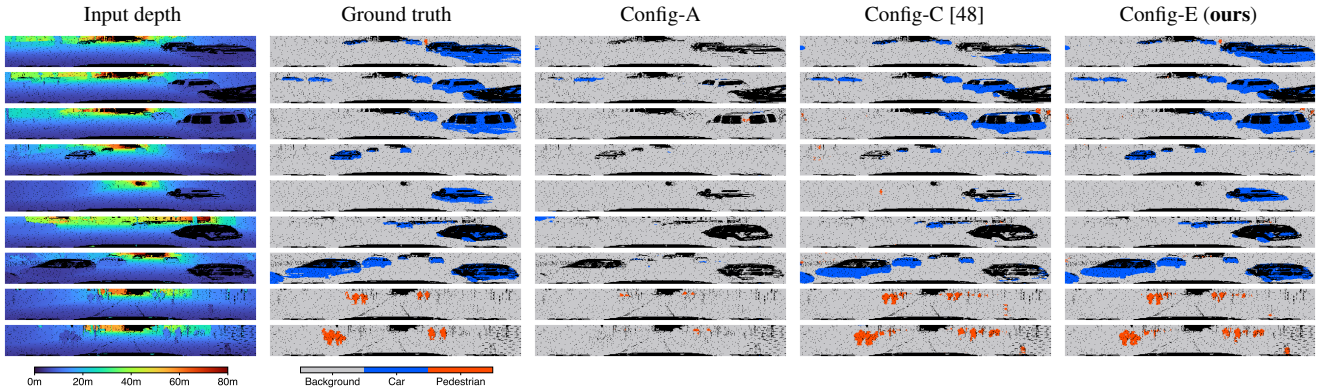


Figure 18. Comparison of Sim2Real segmentation results in 2D range images. We compare three types of ray-drop priors from our main paper. Config-A: GTA-LiDAR without ray-drop rendering. Config-C: GTA-LiDAR with Bernoulli noises from the pixel-wise frequency [47]. Config-E (ours): GTA-LiDAR with Bernoulli noises from our auto-decoded ray-drop probability.