2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

DRUM: Diffusion-based Raydrop-aware Unpaired Mapping for Sim2Real LiDAR Segmentation

Tomoya Miyawaki[1] Kazuto Nakashima[2] Yumi Iwashita[3] Ryo Kurazume[2]

***Abstract*— LiDAR-based semantic segmentation is a key component for autonomous mobile robots, yet large-scale annotation of LiDAR point clouds is prohibitively expensive and time-consuming. Although simulators can provide labeled synthetic data, models trained on synthetic data often underperform on real-world data due to a data-level domain gap. To address this issue, we propose DRUM, a novel Sim2Real translation framework. We leverage a diffusion model pre-trained on unlabeled real-world data as a generative prior and translate synthetic data by reproducing two key measurement characteristics: reflectance intensity and raydrop noise. To improve sample fidelity, we introduce a raydrop-aware masked guidance mechanism that selectively enforces consistency with the input synthetic data while preserving realistic raydrop noise induced by the diffusion prior. Experimental results demonstrate that DRUM consistently improves Sim2Real performance across multiple representations of LiDAR data. The project page is available at `https://miya-tomoya.github.io/drum`.**

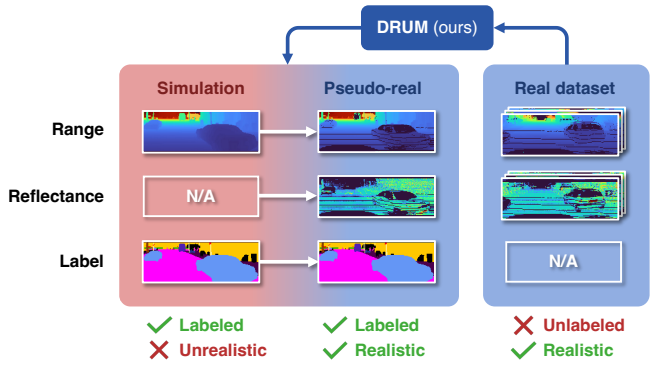


Fig. 1. **Problem formulation**. Given unpaired sets of labeled simulation and unlabeled real samples, our framework DRUM generates labeled pseudo-real samples for training LiDAR segmentation.

I. Introduction

3D LiDAR sensors capture high-fidelity point clouds of the surrounding environment using time-of-flight (ToF) ranging. These LiDAR point clouds have become integral to scene perception for autonomous systems, such as mobile robots and self-driving cars. In particular, semantic segmentation of LiDAR point clouds has been a central task in robotics and computer vision research [1]–[5], which involves assigning a semantic label to every point in a scene. However, the dominant supervised learning paradigm for this task relies heavily on large quantities of *labeled* point clouds, which are prohibitively expensive in terms of manual labor and time. For instance, annotating the widely used SemanticKITTI benchmark [6] required over 1,700 person-hours for its 43,000 scans. This annotation bottleneck motivates research into cost-effective alternatives.

Sim2Real (simulation-to-real) transfer [3, 4, 7, 8] offers a promising solution to the critical annotation bottleneck in LiDAR-based perception. This paradigm leverages simulators equipped with ray-tracing and physics engines to generate vast quantities of automatically labeled synthetic data, circumventing the need for laborious manual annotation. However, models trained exclusively on synthetic data often fail to generalize to real-world data due to the inherent domain gap between simulation and real domains. A major source of this gap stems from two factors governed by both scene properties and sensor characteristics: *reflectance intensity* and *raydrop noise*. Reflectance intensity is the measured strength of the backscattered laser signal, influenced by factors such as surface material, incidence angle, and sensor characteristics. Raydrop noise occurs when the laser signal attenuates or scatters upon hitting a surface, resulting in an insufficient return intensity for detection. It appears as missing pixels in range or reflectance images, as exemplified in Fig. 1. Although existing LiDAR simulators [9]–[11] implement simplified physics models to simulate these effects, they still fall short of capturing the complexity and diversity observed in real-world measurements.

To bridge the domain gap between simulation and the real world, there has been a growing interest in developing machine learning systems to simulate realistic LiDAR data. One prominent line of work is *re-simulation* [12, 13], which synthesizes novel sensor views from an existing set of real-world scans. While this method allows for the reuse of initial annotations, it lacks scalability as it still relies on a manually labeled real-world dataset. Another approach focuses on *Sim2Real translation*, converting labeled synthetic data to the real-world domain. This is often accomplished by building conversion models through self-supervised learning that leverages superimposed multi-scan observations [14, 15] or unsupervised learning with deep generative models [4, 8, 16], such as generative adversarial networks (GANs) [17]. Nevertheless, these methods still struggle to jointly and realistically reproduce the complex patterns of raydrop and reflectance intensity.

*This work was supported by JSPS KAKENHI Grant Number JP23K16974 and JP20H00230.

[1]Tomoya Miyawaki is with the Graduate School of Information Science and Electrical Engineering, Kyushu University, Japan. miyawaki@irvs.ait.kyushu-u.ac.jp

[2]Kazuto Nakashima and Ryo Kurazume are with the Faculty of Information Science and Electrical Engineering, Kyushu University, Japan. k_nakashima@mech.kyushu-u.ac.jp, kurazume@ait.kyushu-u.ac.jp

[3]Yumi Iwashita is with the Jet Propulsion Laboratory, California Institute of Technology, USA. yumi.iwashita@jpl.nasa.gov

To this end, we propose **DRUM** (diffusion-based raydrop-aware unpaired mapping), a novel Sim2Real translation method for semantic segmentation of LiDAR point clouds. The problem formulation is illustrated in Fig. 1. Our method formulates the translation task as posterior sampling of real-world range and reflectance data, conditioned only on the range data from the simulation domain. To represent the prior distribution of the real-world domain, we use diffusion models [18], a family of deep generative models that have garnered significant attention in recent years for their high-fidelity generation capabilities. Although posterior sampling using pre-trained diffusion models has been demonstrated in general restoration tasks [19] including a LiDAR upsampling task [20, 21], we found that inconsistency in raydrop noise between the simulation and real domains causes unstable and unrealistic generation. To address this issue, we introduce two key techniques: a raydrop-aware masked sampling strategy based on a provisional estimate during sampling, and an initialization technique to ensure geometric consistency.

Our contributions can be summarized as follows:

- We propose a novel Sim2Real translation method for LiDAR point clouds. We formulate the task as posterior sampling using unconditional diffusion models pre-trained on the real-world dataset, enabling a unified generation of realistic range, reflectance, and raydrop.
- We propose a raydrop-aware masked guidance mechanism that conditions the generation process on simulation data, thereby stabilizing the diffusion-based posterior sampling.
- We demonstrate that our approach outperforms existing methods in terms of sample fidelity and Sim2Real performance in semantic segmentation.

II. Related Work

Semantic segmentation of LiDAR data is a task of predicting point-level semantic classes, such as cars and roads in driving scenes [6]. Various segmentation models have been proposed primarily using neural networks, depending on the representation of LiDAR data, such as point clouds [2], images [1, 3], and voxels [5]. In many cases, multimodal information, *i.e.*, the measured range and reflectance intensity, is used as input cues for semantic segmentation. Manual annotation is required to construct training data for neural networks, which entails substantial cost and time, as it requires consistent quality for the massive number of 3D points.

Sim2Real transfer is one approach to solving the annotation problem. A variety of simulators leveraging graphics and physics engines, such as CARLA [9], are available, which enable us to automatically collect training data with high-quality annotations. Leveraging these simulators, a number of datasets have been introduced [3, 16], featuring diverse sensor setups and object categories. However, their generalization performance inevitably suffers from a domain gap between simulation and real-world samples. To address these issues, Sim2Real transfer is commonly treated as a domain adaptation problem. Existing approaches are twofold: feature-level mapping and data-level mapping. The feature-level mapping aims to align the distributions between simulation and real domains in a shared feature space, such as through domain-invariant feature learning [22]. The data-level mapping approaches attempt to transform synthetic data to appear more realistic, such as by reproducing raydrop noise [3, 4] and reflectance intensity [16]. Nevertheless, challenges persist with respect to the fidelity of the reproduced modalities and the geometric consistency of the scenes. Furthermore, most prior work addresses these problems in isolation, despite their multimodal nature. We propose a unified framework to estimate the raydrop and reflectance intensity, leveraging the following deep generative models.

Deep generative models have recently emerged as a powerful framework for capturing the distributions of a wide variety of data, including images and videos. These models can synthesize novel data by sampling from a learned distribution. Furthermore, the trained model can be used as a data prior for downstream tasks such as restoration and editing. Numerous generative methods have also been developed for the LiDAR domain [8, 15, 20, 21, 23]–[27] based on variational autoencoders (VAEs) [28], generative adversarial networks (GANs) [17], and diffusion models [18, 29]. In particular, approaches using diffusion models [15, 20, 21, 26, 27] have significantly improved the quality of generated data and the stability of model training against other approaches. Diffusion models describe the data generation process using stochastic or ordinary differential equations (SDE/ODE), which are guided by the gradient of the time-evolving data distribution, known as the *score*. This formulation allows the models to be adapted for conditional generation by modifying the score via Bayesian inference, which has led to their application in downstream tasks such as upsampling and completion. In this paper, we repurpose the power of diffusion models for the task of Sim2Real domain adaptation of LiDAR data.

III. Method

A. Problem Formulation

Given an unpaired set consisting of a *labeled* simulation dataset $\mathcal{D}_{\mathrm{sim}}$ and an *unlabeled* real dataset $\mathcal{D}_{\mathrm{real}}$, we aim to address the problem of unpaired Sim2Real translation, as illustrated in Fig. 1. More specifically, $\mathcal{D}_{\mathrm{sim}}$ contains only the *range* modality, whereas $\mathcal{D}_{\mathrm{real}}$ contains both *range* and *reflectance* modalities. Furthermore, the real samples exhibit raydrop noise, which manifests as missing pixels in the images.

Data representation. Since the majority of LiDAR generative models [8, 20, 21, 23, 25]–[27] have been developed based on the image representation, we also perform the data translation on the same image space. We assume a spinning multi-beam LiDAR with angular resolution W in azimuth and H in elevation that measures range and reflectance at each angle (θ, ϕ). These $H \times W$ measurements are projected onto a two-channel equirectangular image space $\mathbb{R}^{2\times H\times W}$. The simulation data contain only range modality, and the reflectance channel is zero-padded. Here, we first formulate

the absence of the reflectance channel as a linear degradation of the sample from the real domain to the simulation domain, and then consider the following linear inverse problem to recover a real-domain sample.

Linear inverse problem. A general form of a linear degradation process can be written as the following observation equation:

$$\boldsymbol{y} = H\boldsymbol{x} + \boldsymbol{n}, \tag{1}$$

where $\boldsymbol{y} \in \mathbb{R}^m$ is an observation, $H \in \mathbb{R}^{m \times n}$ is a known measurement matrix, $\boldsymbol{x} \in \mathbb{R}^n$ is an original signal, and $\boldsymbol{n} \in \mathbb{R}^m$ is the observation noise drawn from $\mathcal{N}(0, \sigma^2 \boldsymbol{I})$. The goal of the linear inverse problem is to estimate $\boldsymbol{x}$ from the observed data $\boldsymbol{y}$.

Our approach in LiDAR Sim2Real. Instantiating H as a corruption matrix that zeroes the reflectance channel, our task can be defined as estimation of $\boldsymbol{x}$ in real domain that contains both range and reflectance from $\boldsymbol{y}$ in simulation domain that contains only range information. We note that, while this formulation defines the *forward* corruption of reflectance on $\boldsymbol{y} \leftarrow \boldsymbol{x}$, it cannot account for the non-linear effects of raydrop noise, *i.e.*, *backward* corruption on $\boldsymbol{y} \rightarrow \boldsymbol{x}$. Therefore, we formulate Sim2Real translation as a hybrid inference problem: explicitly solving a linear inverse problem for reflectance restoration with the known H, while implicitly modeling the non-linear raydrop degradation through our proposed mechanism, as detailed in Sec. III-C.

B. Diffusion Posterior Sampling

In this section, we describe a method for solving the aforementioned linear inverse problem through posterior sampling $\boldsymbol{x} \sim p(\boldsymbol{x} \mid \boldsymbol{y})$ with diffusion models. We first review the unconditional sampling using a diffusion model [29], describe how we can solve the general linear inverse problems [19] by leveraging the same pre-trained diffusion model, and then identify the challenges in our case.

Unconditional sampling. Diffusion models formulate an unconditional sampling of the data distribution $\boldsymbol{x} \sim p(\boldsymbol{x})$ based on the following reverse-time SDE [29]:

$$d\boldsymbol{x}_t = \left[f(\boldsymbol{x}_t, t) - g^2(t) \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t) \right] dt + g(t) d\bar{\boldsymbol{w}}, \tag{2}$$

where the state variable $\boldsymbol{x}_t$ evolves over the continuous time $t \in [1, 0]$, $f(\boldsymbol{x}_t, t)$ and $g(t)$ are coefficient functions designed according to the types of diffusion models, $\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t)$ is a learnable score function modeled by a neural network, and $\bar{\boldsymbol{w}}$ is the standard Wiener process. We can sample data by integrating Eq. (2), initializing with a Gaussian sample at $t = 1$. One can approximate Eq. (2) by deterministic samplers such as probability flow ODE [29] and DDIM [30].

Posterior sampling. Linear inverse problems can also be addressed within a similar framework. To sample $\boldsymbol{x}$ conditioned on $\boldsymbol{y}$, we aim to draw samples from the posterior distribution $p(\boldsymbol{x} \mid \boldsymbol{y})$. To this end, we can replace the score in Eq. (2) with the following conditional score:

$$\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t \mid \boldsymbol{y}) = \underbrace{\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t)}_{\text{prior}} + \underbrace{\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{y} \mid \boldsymbol{x}_t)}_{\text{guidance}}, \tag{3}$$

where we use the Bayes' rule: $p(\boldsymbol{x} \mid \boldsymbol{y}) \propto p(\boldsymbol{x}) p(\boldsymbol{y} \mid \boldsymbol{x})$. It is worth noting that the first *prior* term is the unconditional score that can be trained solely $\boldsymbol{x}$ based on Eq. (2). Moreover, the second *likelihood* term is referred to as *guidance*, which can be approximated from Eq. (1). We employ the definition of ΠGDM [19] as follows:

$$\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{y} \mid \boldsymbol{x}_t) \approx r_t^{-2} \left[\left(H^\dagger \boldsymbol{y} - H^\dagger H \hat{\boldsymbol{x}}_t \right)^\top \frac{\partial \hat{\boldsymbol{x}}_t}{\partial \boldsymbol{x}_t} \right]^\top, \tag{4}$$

where $\hat{\boldsymbol{x}}_t$ is a tentative estimate of $\boldsymbol{x}_0$ given $\boldsymbol{x}_t$ using Tweedie's formula [31], $H^\dagger$ is a Moore-Penrose pseudoinverse matrix of H, and r_t is a time-dependent standard deviation that scales the gradient according to the noise level at timestep t.

Challenges. Based on the above formulation, the pseudoinverse guidance in Eq. (4) computes a likelihood gradient at each time step, driven by the sim–real discrepancy $H^\dagger \boldsymbol{y} - H^\dagger H \hat{\boldsymbol{x}}_t$. This guidance enforces strong consistency with the observation $\boldsymbol{y}$ and promotes the inpainting of missing content, ensuring that the samples conform to the observation. However, applying it directly in our task is problematic. Uniformly evaluating the discrepancy across all pixels drives the reconstruction $\hat{\boldsymbol{x}}_t$ to replicate the clean simulation $\boldsymbol{y}$. Since $\boldsymbol{y}$ contains no raydrops, the guidance suppresses the realistic drop patterns that the diffusion model begins to form, resulting in "over-inpainted" and degraded samples.

C. Raydrop-aware Masked Guidance

To avoid the above failure mode, we selectively gate the guidance in sampling, so that the raydrop pixels induced by the prior term are not overwritten by the simulation input $\boldsymbol{y}$. Fig. 2 illustrates the overall framework of our posterior sampling and Fig. 3 shows the generated samples.

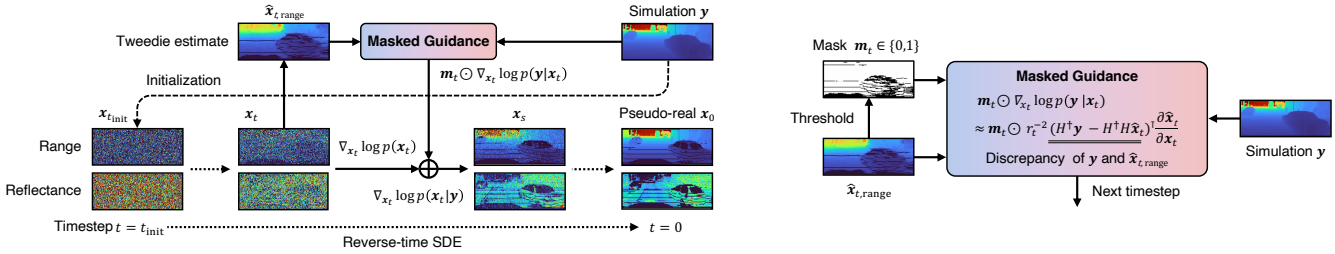
Progressive masking. The core idea is to modulate the guidance term of Eq. (3) with a *raydrop-aware mask* $\boldsymbol{m}_t$ estimated at each timestep t as follows:

$$\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t \mid \boldsymbol{y}) = \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t) + \boldsymbol{m}_t \odot \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{y} \mid \boldsymbol{x}_t), \tag{5}$$

where $\odot$ denotes the element-wise product. Similar to the guidance computation, we leverage the Tweedie estimate $\hat{\boldsymbol{x}}_t$ to construct $\boldsymbol{m}_t$ at each timestep t. Since the diffusion model is pre-trained on real LiDAR scans that contain raydrop, the Tweedie estimate $\hat{\boldsymbol{x}}_t$ also reflects plausible raydrop patterns based on the learned prior distribution (see Fig. 2 for instance). Therefore, we can construct a raydrop-aware mask $\boldsymbol{m}_t$ by thresholding the range channel of $\hat{\boldsymbol{x}}_t$, whose i-th element is determined as follows:

$$m_t^i = \begin{cases} 1 & \text{if } \hat{x}^i_{t,\text{range}} > \eta \\ 0 & \text{otherwise} \end{cases}, \tag{6}$$

where $\hat{x}^i_{t,\text{range}}$ is the estimated range value of the i-th pixel at the timestep t, and η is a threshold value. Accordingly, the guidance enforces consistency with the synthetic input $\boldsymbol{y}$ only at valid-return pixels ($m_t^i = 1$), while raydrop pixels ($m_t^i = 0$) are left unaffected, allowing the diffusion prior to freely maintain the learned raydrop patterns. As shown



(a) Posterior sampling using a pre-trained diffusion model

(b) Masked guidance to condition with the simulation range $\boldsymbol{y}$

Fig. 2. **Overview of Sim2Real translation by DRUM**. (a) The unconditional generation process of diffusion models is conditioned by the masked guidance with the simulation sample $\boldsymbol{y}$. (b) In masked guidance, we first generate the raydrop-aware $\boldsymbol{m}_t$ mask from the tentative Tweedie sample $\hat{\boldsymbol{x}}_t$ and then compute the sim–real discrepancy based on the pseudoinverse method [19]. The operator H corrupts the reflectance modality.

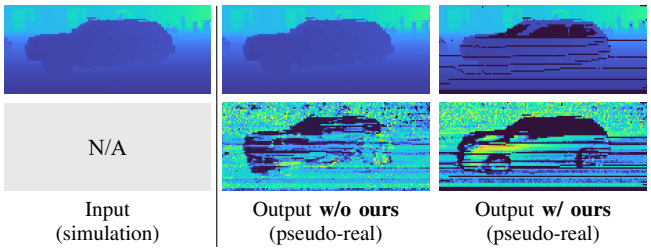


Fig. 3. **Ablation of our masked guidance**. We compare the range (top row) and reflectance (bottom row) pseudo-real samples produced with and without our raydrop-aware masked guidance. Our method successfully reproduces the raydrop noise on the car, as well as the reflectance modality.

in Fig. 3, our approach prevents over-inpainting and successfully reproduces realistic raydrop noise and reflectance intensity.

Initialization. The state variable $\boldsymbol{x}_t$ of Eq. (2) evolves from Gaussian noise $\boldsymbol{x}_1$ during sampling. Thus, the Tweedie sample $\hat{\boldsymbol{x}}_t$ and its derived $\boldsymbol{m}_t$ may not be aligned with the target simulation $\boldsymbol{y}$ in the early steps, leading to unstable progression. To mitigate this issue, we initialize the low-frequency component of evolving sample $\boldsymbol{x}_t$ based on $\boldsymbol{y}$, following a strategy similar to SDEdit [32]. Specifically, we add noise to the simulation input $\boldsymbol{y}$ according to the forward SDE [29] up to time t_{init}. The resulting noisy latent $\boldsymbol{x}_{t_{\text{init}}}$ serves as the starting point of the reverse diffusion process. We illustrate this step in Fig. 2(a).

Harmonization. Applying masked guidance can create unnatural seams at the boundaries between the guided regions (where $m_t^i = 1$) and the unconditionally generated regions (where $m_t^i = 0$). This is because the two areas are constrained by different dynamics. To harmonize these distinct areas, we employ the iterative resampling technique introduced in RePaint [33]. For each timestep t in the guided phase ($t < t_{\text{init}}$), a single reverse step is replaced by a refinement loop of unconditional reverse and forward SDE steps. This iterative procedure allows information to propagate naturally between the two regions, ensuring that they are seamlessly blended into a coherent and realistic output.

D. Architecture of Diffusion Prior Models

For the prior term of Eq. (5), we build an unconditional diffusion model over 2-channel LiDAR range–reflectance images. The backbone is an encoder–decoder neural network (U-Net [34]) conditioned on the timestep information; the network takes the state variable $\boldsymbol{x}_t$ and a log-SNR embedding as input. We adopt the standard $\boldsymbol{\epsilon}$-prediction re-parameterization [18]; an involved Gaussian noise $\boldsymbol{\epsilon}$ is predicted by the neural network $\epsilon_\theta(\boldsymbol{x}_t, t)$, rather than directly estimating the score. We denote the unconditional score (as in Sec. III) by $s_\theta(\boldsymbol{x}_t, t) \equiv \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t)$. Under the variance-preserving parameterization of SDE [29] with $p(\boldsymbol{x}_t \mid \boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t \mid \alpha_t \boldsymbol{x}_0, \sigma_t^2 \boldsymbol{I})$, the score admits a closed-form conversion from $\boldsymbol{\epsilon}$-prediction:

$$s_\theta(\boldsymbol{x}_t, t) = -\frac{\epsilon_\theta(\boldsymbol{x}_t, t)}{\sigma_t}, \tag{7}$$

and we can obtain the Tweedie sample $\hat{\boldsymbol{x}}_t$ by

$$\hat{\boldsymbol{x}}_t = \frac{\boldsymbol{x}_t - \sigma_t \, \epsilon_\theta(\boldsymbol{x}_t, t)}{\alpha_t}, \tag{8}$$

where α_t and σ_t are the noise schedule coefficients, which can be calculated from $f(\boldsymbol{x}_t, t)$ and $g(t)$ of Eq. (2). For numerical stability, we clip $\hat{\boldsymbol{x}}_t$ to a fixed range.

E. Training Segmentation Models

Finally we train segmentation models with the generated pseudo-real samples. From the aforementioned posterior sampling, we obtain the sample $\boldsymbol{x}_0$ translated from $\boldsymbol{y}$. To ensure consistency with the annotated labels of the simulation input $\boldsymbol{y}$, we re-initialize the valid range values of $\boldsymbol{x}_0$ with $\boldsymbol{y}$ while keeping the generated raydrop noise. Similar to Eq. (6), we first extract a raydrop mask $\boldsymbol{m}_0$ from $\boldsymbol{x}_{0,\text{range}}$. We then apply $\boldsymbol{m}_0$ to the synthetic range $\boldsymbol{y}_{\text{range}}$ and the generated reflectance $\boldsymbol{x}_{0,\text{reflect}}$. Both are then concatenated for training segmentation models. For training point-based and voxel-based backbones [2, 5], we convert the pseudo-real sample into 3D point clouds and sparse voxels, respectively.

IV. Experiments

In this section, we evaluate our approach in terms of sample fidelity (Sec. IV-B) and Sim2Real performance of semantic segmentation (Sec. IV-C).

TABLE I
TWO SCENARIOS IN OUR EXPERIMENTS

Dataset	Domain	#Classes	#Samples	Resolution
GTA-LiDAR [3]	Simulation	2	121,087	64×512
KITTI Raw [35]	Real		10,848	
SynLiDAR [16]	Simulation	32†	198,396	64×1024
SemanticKITTI [6]	Real	25†	43,552	

† We use the common 19 classes as listed in Tab. IV.

A. Settings

Datasets and tasks. The datasets used in our experiments are summarized in Tab. I. For the *sample quality* evaluation, we use the SynLiDAR dataset [16] as a simulation dataset and SemanticKITTI dataset [6] as a real dataset. We first train a diffusion model on the real dataset, translate the simulation dataset into the pseudo-real dataset, and then evaluate the distributional similarity between the real and pseudo-real samples. For the *semantic segmentation* evaluation, we conduct two scenarios: (1) Following Wu *et al.* [3], we train 2-class segmentation models on the GTA-LiDAR dataset [3] and evaluate on the 90° frontal subset [3] of KITTI dataset [35], referred to as KITTI-frontal. (2) We also conduct 19-class semantic segmentation. We use SynLiDAR dataset [16] for training and SemanticKITTI dataset [6] for evaluation on the common 19 classes.

Implementation details. For building diffusion priors of real data, we employ R2DM [20], a continuous-time diffusion model of range/reflectance images. To improve generation quality and fidelity, we increase the model capacity of the official implementation[1] from 31M to 285M parameters by doubling the number of channels. Pseudo-real samples are generated via posterior sampling as in Sec. III. We set $t_{\text{init}} = 0.8$ for initialization. Similar to ΠGDM [19], we use DDIM sampler [30] for Eq. (2) with discretization of 32 uniform steps, each with our masked guidance of Eq. (5). The progressive mask $\boldsymbol{m}_t$ is built from the Tweedie estimate by thresholding the normalized range in $[-1, 1]$ using $\eta = -0.3$. For the harmonization process, we perform three additional resampling cycles for each sampling step. The entire pipeline requires approximately 8 seconds per sample on a single NVIDIA RTX 6000 Ada GPU.

B. Sample Fidelity

Baselines. We compare four methods for simulating reflectance intensity and raydrop noise, as listed in Tab. II. The rendering model in Tab. II represents the officially-provided SynLiDAR samples[2], where the reflectance intensity is generated using the voxel-based supervised model [16]. For diffusion-based approaches, we compare three methods using different components of our framework: SDEdit [32] which initializes the sampling at t_{init}, ΠGDM [19] which uses the original pseudoinverse guidance without our progressive mask $\boldsymbol{m}_t$, and our DRUM which integrates all components. All diffusion methods use the same unconditional R2DM model pre-trained on SemanticKITTI [6].

Evaluation metrics. To evaluate the sample fidelity, we compute the distributional similarity between real and pseudo-real samples across multiple levels of data representation. We use five evaluation metrics that extend the Fréchet Inception distance (FID) for LiDAR data: Fréchet range distance (FRD) [21], Fréchet range image distance (FRID) [27], Fréchet point cloud distance (FPD) [36], Fréchet point-based volume distance (FPVD) [27], and Fréchet sparse volume distance (FSVD) [27]. FRD and FRID evaluate similarity in range image format, FPD in point cloud format, and FPVD and FSVD in voxel format. FRD uses both range and reflectance channels for evaluation, while FRID focuses solely on the range channel. The point cloud and voxel representations are derived from the range image through spherical projection and voxelization, respectively. Tab. II summarizes the correspondence between metrics and data representations.

Quantitative results. Tab. II presents the quantitative results of the fidelity evaluation. Our proposed method significantly improves the fidelity of SynLiDAR samples [16] constructed by the simulator. However, our scores are close to those of SDEdit, which only sets the initial value of the reverse diffusion process. We attribute these results to the limitations of the evaluation, since fidelity metrics quantify distributional similarity, not per-sample quality. We further discuss the difference between our method and SDEdit in the qualitative results.

Qualitative results. In Fig. 4, we compare the pseudo-real samples generated by different methods. SDEdit alone generates realistic reflectance intensity and raydrop noise, but fails to preserve the scene content from the simulation data, such as vehicles and the cityscape, as highlighted by the yellow circles. Conversely, ΠGDM alone preserves the scene content but fails to reproduce raydrop noise due to the strong guidance from the likelihood term. In contrast, our method successfully generates realistic reflectance intensity and raydrop noise while maintaining high fidelity to the original scene content. Notably, our approach accurately reproduces the raydrop noise around car windows, which can be seen frequently in real samples due to the reflective properties of glass.

C. Semantic Segmentation Results

Baselines. For the first scenario, we follow the protocol of prior work [8] to evaluate different methods of rendering raydrop noise, as listed in Table III. We compare against several methods: no rendering (config-A), global statistics (config-B), spatial statistics [3] (config-C), ePointDA [4] (config-D), DUSty [25] (config-E), and DUSty v2 [8] (config-F). The global/spatial statistics methods are Bernoulli sampling based on the average ratio of raydrop pixels. The DUSty methods are based on the raydrop probability map generated by GANs. ePointDA is an unpaired mapping model trained to predict the raydrop noise using CycleGAN [37]. Our proposed diffusion-based approach is denoted as config-G.

[1] https://github.com/kazuto1011/r2dm
[2] https://github.com/xiaoaoran/SynLiDAR

TABLE II

QUANTITATIVE COMPARISON OF SIM2REAL METHODS ON SAMPLE FIDELITY.

Method	Raydrop	Reflectance	FRD ↓	FRID ↓	FPD ↓	FPVD ↓	FSVD ↓
Rendering model [16]		✓	2198.8	76.6	148.1	60.3	69.9
SDEdit [32]	✓	✓	**867.6**	40.2	140.1	55.1	63.4
ΠGDM [19]	✓	✓	1413.9	75.0	146.1	59.8	69.4
DRUM (ours)	✓	✓	921.0	**39.6**	**139.4**	**54.2**	**62.5**

Notation: range image, reflectance image, point cloud, voxel. The best score is in **bold** and the top two scores are shaded.

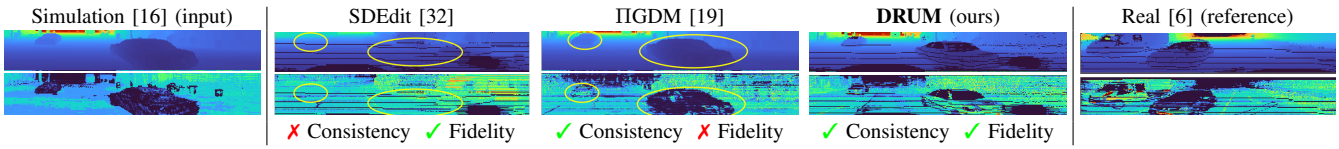


Fig. 4. **Qualitative comparison of pseudo-real samples.** We compare the range (top row) and reflectance (bottom row) samples produced by different methods. The reflectance input is from the rendering model [16] while we do not use it for producing the pseudo-real samples. Our method shows better results in terms of consistency to the input and fidelity comparable to the reference.

TABLE III

QUANTITATIVE COMPARISON OF SIM2REAL METHODS ON SEMANTIC SEGMENTATION (GTA-LIDAR → KITTI-FRONTAL)

Config	Method	IoU (%, ↑) Car	Pedestrian	Mean
A	No rendering	1.1	2.4	1.7
B	Global statistics	55.2	25.1	40.2
C	Spatial statistics [3]	59.0	22.5	40.7
D	ePointDA [4]	66.2	24.8	45.5
E	DUSty [25]	59.1	**28.0**	43.5
F	DUSty v2 [8]	**67.3**	25.2	46.3
G	**DRUM** (ours)	66.5	26.9	**46.7**

We use SqueezeSegV2 [3] for the semantic segmentation architecture across all configurations. In our second scenario, we evaluate the effectiveness of our approach across different representations of LiDAR data on the 19-class semantic segmentation task. We train segmentation models on range images, point clouds, and voxels. We employ RangeNet [1], RandLA-Net [2], and MinkowskiNet [5], respectively.

Quantitative results. The results for the two experimental scenarios are presented in Tabs. III and IV, respectively. In the first 2-class scenario (Tab. III), our diffusion-based approach (config-G) achieves the best performance with an mIoU of 46.7%, surpassing all baseline methods. In the second 19-class scenario (Tab. IV), our method consistently improves performance across all data representations. A key factor in this success is the realistic modeling of raydrop. Compared to the baseline rendering, our DRUM model using only raydrop yields substantial gains across all representations, with image-based models showing a more than a twofold increase in mIoU. This improvement is particularly notable in the car class. As shown in our fidelity evaluation (Fig. 4), our method successfully reproduces the distinct raydrop patterns around car windows, which frequently arise in real LiDAR scans due to reflections from glass surfaces. In contrast, the contribution of reflectance is more nuanced. We hypothesize that subtle inconsistencies between the range and reflectance modalities can offset the potential benefits of multimodality. Addressing this issue to better exploit both modalities remains an important direction for future work.

Qualitative results. Fig. 5 presents a qualitative comparison of image-based segmentation results on the SemanticKITTI validation set. Compared to the baseline model trained on the official SynLiDAR samples [16], our method produces results that are visually closer to the ground truth. In particular, background classes such as *vegetation* and *building* are more consistently recognized, reducing the spurious regions observed in the baseline. For foreground objects, the baseline often struggles to disentangle cars from their surroundings, resulting in blurred boundaries and cases where cars are partially merged with vegetation or nearby structures. By contrast, our method better preserves *cars*, yielding clearer object boundaries and fewer false negatives.

Mixed training. Given the notable improvements achieved by our method on image-based 19-class semantic segmentation, we further evaluate a mixed training scenario with a limited labeled real dataset. To assess the impact of different levels of data availability, we systematically sample labeled subsets from SemanticKITTI at ratios of 1%, 20%, 50%, and 100%. We then train RangeNet [1] using mini-batches that are equally sampled from real and pseudo-real data. The results are shown in Fig. 6. Training with real data alone achieves a validation mIoU of 36.7%. Using only 20% of the real data combined with pseudo-real samples improves the mIoU to 37.1%, already surpassing the real-only baseline. When all real data are combined with pseudo-real samples, the mIoU further increases to 39.2%. Notably, we observe a substantial improvement in minority classes. While real-only training achieves an IoU of 5.2% for the person class, mixed training improves this to 20.5%. An IoU of the bicyclist class was also improved from 28.8% to 51.1%. This improvement is clearly reflected in the qualitative results shown in Fig. 7.

V. CONCLUSIONS

In this paper, we introduced DRUM, a novel Sim2Real translation framework for LiDAR semantic segmentation. By formulating domain adaptation as posterior sampling

TABLE IV

QUANTITATIVE COMPARISON OF SIM2REAL METHODS ON SEMANTIC SEGMENTATION (SYNLIDAR → SEMANTICKITTI).

Network	Sim2Real method	Raydrop	Reflectance	Intersection-over-Union (IoU, %) ↑ Car	Bicycle	Motorcycle	Truck	Bus	Person	Bicyclist	Motorcyclist	Road	Parking	Sidewalk	Other-ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic-sign	Mean (mIoU)
Image-based [1]	–			6.2	2.0	0.4	2.1	1.1	3.3	5.8	0.0	8.0	0.5	6.5	0.0	28.6	2.3	**49.0**	13.7	20.3	12.9	**2.2**	8.7
	Rendering model [16]		✓	3.4	**3.1**	0.4	**4.9**	1.4	3.0	3.3	0.0	5.8	0.2	8.3	0.0	30.8	3.6	33.9	15.3	21.6	12.7	1.5	8.1
	DRUM (ours)	✓		56.2	2.7	5.9	3.8	**7.1**	**6.7**	**19.3**	0.0	49.3	**7.4**	30.2	0.0	35.2	**4.6**	31.7	**18.9**	19.3	**17.5**	2.1	16.7
	DRUM (ours)	✓	✓	**57.3**	1.9	**8.6**	3.1	5.5	5.8	14.0	0.0	**52.8**	5.9	**32.6**	0.0	**40.3**	**4.1**	40.7	18.6	**25.3**	14.6	1.9	**17.5**
Point-based [2]	–			29.6	5.8	2.1	1.4	3.4	13.0	29.0	0.1	12.3	2.1	30.6	0.0	37.1	5.5	61.0	23.1	38.0	35.5	12.0	18.0
	Rendering model [16]		✓	32.2	1.2	**11.6**	**2.1**	**4.3**	16.1	31.5	0.0	9.3	4.3	28.7	0.0	54.7	3.8	**65.8**	14.2	15.9	19.2	8.0	17.0
	DRUM (ours)	✓		40.5	**16.4**	6.6	0.9	0.6	18.0	**51.1**	0.3	**23.0**	9.6	**32.1**	0.0	58.5	10.3	64.1	24.6	**38.6**	38.5	8.8	23.3
	DRUM (ours)	✓	✓	**57.4**	6.6	5.9	1.0	**12.0**	**19.3**	36.1	**0.8**	16.8	**12.5**	30.9	0.0	**61.7**	**17.9**	61.4	**30.5**	33.0	**40.6**	**12.1**	**24.0**
Voxel-based [5]	–			55.5	10.3	25.5	1.4	2.7	17.6	50.7	2.8	12.5	**7.6**	32.3	0.0	29.9	**19.0**	62.4	21.1	37.7	11.5	1.9	21.2
	Rendering model [16]		✓	61.0	8.3	28.9	1.4	2.5	16.0	53.2	1.7	15.8	4.2	32.0	0.0	30.5	9.1	62.4	16.5	35.0	7.4	2.2	20.4
	DRUM (ours)	✓		73.1	20.2	28.5	**2.4**	4.9	20.2	55.5	**5.8**	**17.7**	7.0	**34.8**	**0.1**	47.2	13.6	67.9	29.4	**52.8**	23.5	4.4	26.8
	DRUM (ours)	✓	✓	**76.3**	**23.3**	**33.2**	1.3	**11.4**	**31.1**	**57.2**	5.6	16.6	7.0	33.9	0.0	**59.1**	9.4	**72.0**	**34.3**	52.0	**29.7**	**5.9**	**29.4**

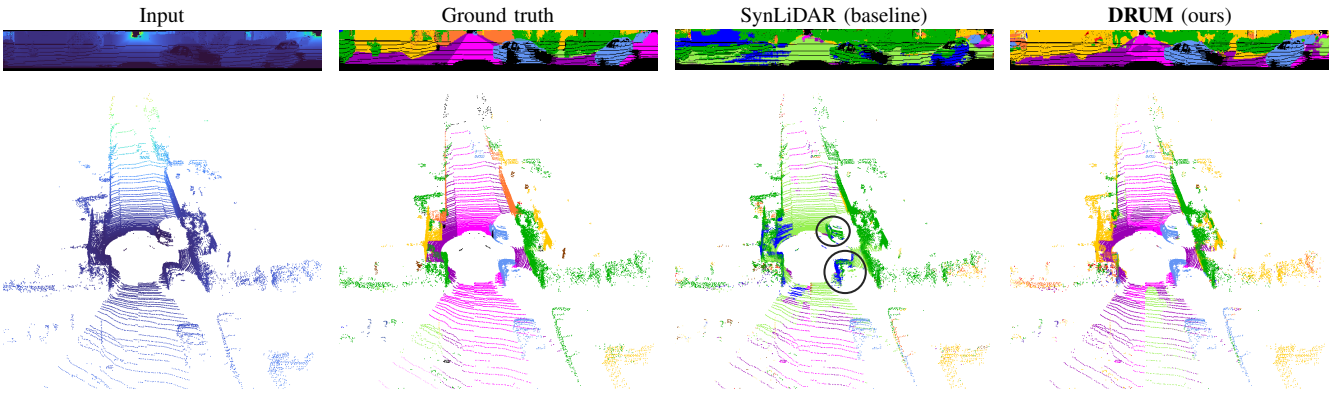


Fig. 5. **Qualitative comparison of semantic segmentation results.** We show the results of the image-based method on the representation of images (top row) and point clouds (bottom row). Our approach significantly improves car detection (black circles) compared to the baseline.

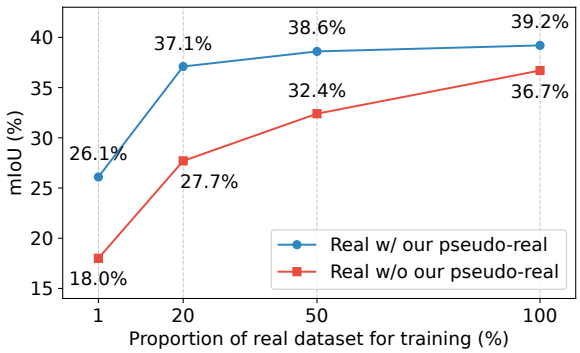


Fig. 6. **Effect of mixed training using our pseudo-real samples.** We show scores of mIoU (%) on image-based 19-class semantic segmentation. We mix real data at ratios of 1%, 20%, 50%, and 100% with our pseudo-real data during training.

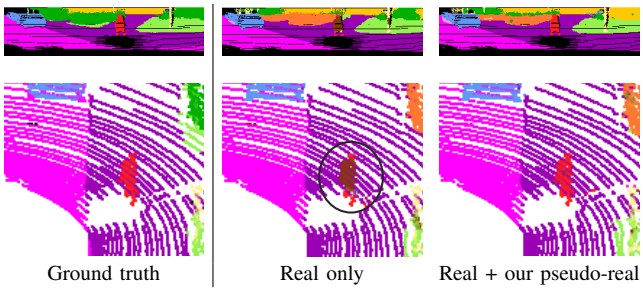


Fig. 7. **Qualitative comparison of segmentation results in the mixed training scenario.** When trained solely on real data (middle), the model misclassifies the person as the trunk.

using diffusion models, DRUM generates realistic pseudo-real data that capture complex measurement characteristics such as reflectance intensity and raydrop noise. Our raydrop-aware masked guidance mechanism preserves geometric consistency while enabling faithful translation of domain-specific features. Our experiments demonstrated that DRUM outperforms existing methods in terms of sample fidelity and consistently improves semantic segmentation performance across image, point cloud, and voxel representations. In future work, we plan to explore emerging flow matching models as potential alternatives to diffusion priors, aiming for faster and more flexible Sim2Real adaptation.

REFERENCES

[1] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet++: Fast and accurate LiDAR semantic segmentation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4213–4220, 2019.

[2] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11108–11117, 2020.

[3] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. 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)*, pp. 4376–4382, 2019.

[4] S. Zhao, Y. Wang, B. Li, B. Wu, Y. Gao, P. Xu, T. Darrell, and K. 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*, vol. 35, pp. 3500–3509, 2021.

[5] C. Choy, J. Gwak, and S. Savarese, "4D spatio-temporal convnets: Minkowski convolutional neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3075–3084, 2019.

[6] 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)*, pp. 9297–9307, 2019.

[7] B. Wu, A. Wan, X. Yue, and K. 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)*, pp. 1887–1893, 2018.

[8] K. Nakashima, Y. Iwashita, and R. Kurazume, "Generative range imaging for learning scene priors of 3D LiDAR data," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 1256–1266, 2023.

[9] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the Conference on Robot Learning (CoRL)*, pp. 1–16, 2017.

[10] NVIDIA Corporation, "Isaac Sim." `https://github.com/isaac-sim/IsaacSim`. version 5.0.0.

[11] W. Jansen, E. Verreycken, A. Schenck, J.-E. Blanquart, C. Verhulst, N. Huebel, and J. Steckel, "Cosys-AirSim: A real-time simulation framework expanded for complex industrial applications," in *Proceedings of the Annual Modeling and Simulation Conference (ANNSIM)*, pp. 37–48, 2023.

[12] S. Manivasagam, S. Wang, K. Wong, W. Zeng, M. Sazanovich, S. Tan, B. Yang, W.-C. Ma, and R. Urtasun, "LiDARsim: Realistic LiDAR simulation by leveraging the real world," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11167–11176, 2020.

[13] J. Zhang, F. Zhang, S. Kuang, and L. Zhang, "Nerf-lidar: Generating realistic lidar point clouds with neural radiance fields," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, pp. 7178–7186, 2024.

[14] B. Guillard, S. Vemprala, J. K. Gupta, O. Miksik, V. Vineet, P. Fua, and A. Kapoor, "Learning to simulate realistic LiDARs," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8173–8180, 2022.

[15] V. Zyrianov, H. Che, Z. Liu, and S. Wang, "Lidardm: Generative lidar simulation in a generated world," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6055–6062, 2025.

[16] A. Xiao, J. Huang, D. Guan, F. Zhan, and S. Lu, "Transfer learning from synthetic to real LiDAR point cloud for semantic segmentation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, pp. 2795–2803, 2022.

[17] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 2672–2680, 2014.

[18] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 6840–6851, 2020.

[19] J. Song, A. Vahdat, M. Mardani, and J. Kautz, "Pseudoinverse-guided diffusion models for inverse problems," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.

[20] K. Nakashima and R. Kurazume, "LiDAR data synthesis with denoising diffusion probabilistic models," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14724–14731, 2024.

[21] V. Zyrianov, X. Zhu, and S. Wang, "Learning to generate realistic LiDAR point clouds," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 17–35, 2022.

[22] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. March, and V. Lempitsky, "Domain-adversarial training of neural networks," *Journal of Machine Learning Research (JMLR)*, vol. 17, no. 59, pp. 1–35, 2016.

[23] L. Caccia, H. van Hoof, A. Courville, and J. Pineau, "Deep generative modeling of LiDAR data," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5034–5040, 2019.

[24] Y. Xiong, W.-C. Ma, J. Wang, and R. Urtasun, "Learning compact representations for lidar completion and generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1074–1083, 2023.

[25] K. Nakashima and R. Kurazume, "Learning to drop points for LiDAR scan synthesis," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 222–229, 2021.

[26] K. Nakashima, X. Liu, T. Miyawaki, Y. Iwashita, and R. Kurazume, "Fast LiDAR data generation with rectified flows," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10057–10063, 2025.

[27] H. Ran, V. Guizilini, and Y. Wang, "Towards realistic scene generation with LiDAR diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[28] D. Kingma, T. Salimans, B. Poole, and J. Ho, "Variational diffusion models," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, pp. 21696–21707, 2021.

[29] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[30] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[31] C. M. Stein, "Estimation of the mean of a multivariate normal distribution," *The annals of Statistics*, pp. 1135–1151, 1981.

[32] C. Meng, Y. He, Y. Song, J. Song, J. Wu, J.-Y. Zhu, and S. Ermon, "SDEdit: Guided image synthesis and editing with stochastic differential equations," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

[33] A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, and L. Van Gool, "RePaint: Inpainting using denoising diffusion probabilistic models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11461–11471, 2022.

[34] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 234–241, 2015.

[35] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research (IJRR)*, vol. 32, no. 11, pp. 1231–1237, 2013.

[36] D. W. Shu, S. W. Park, and J. 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)*, pp. 3859–3868, 2019.

[37] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2223–2232, 2017.