2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

LiDAR Data Synthesis with Denoising Diffusion Probabilistic Models

Kazuto Nakashima[1] and Ryo Kurazume[1]

***Abstract*— Generative modeling of 3D LiDAR data is an emerging task with promising applications for autonomous mobile robots, such as scalable simulation, scene manipulation, and sparse-to-dense completion of LiDAR point clouds. While existing approaches have demonstrated the feasibility of image-based LiDAR data generation using deep generative models, they still struggle with fidelity and training stability. In this work, we present R2DM, a novel generative model for LiDAR data that can generate diverse and high-fidelity 3D scene point clouds based on the image representation of range and reflectance intensity. Our method is built upon denoising diffusion probabilistic models (DDPMs), which have shown impressive results among generative model frameworks in recent years. To effectively train DDPMs in the LiDAR domain, we first conduct an in-depth analysis of data representation, loss functions, and spatial inductive biases. Leveraging our R2DM model, we also introduce a flexible LiDAR completion pipeline based on the powerful capabilities of DDPMs. We demonstrate that our method surpasses existing methods in generating tasks on the KITTI-360 and KITTI-Raw datasets, as well as in the completion task on the KITTI-360 dataset. Our project page can be found at `https://kazuto1011.github.io/r2dm`.**

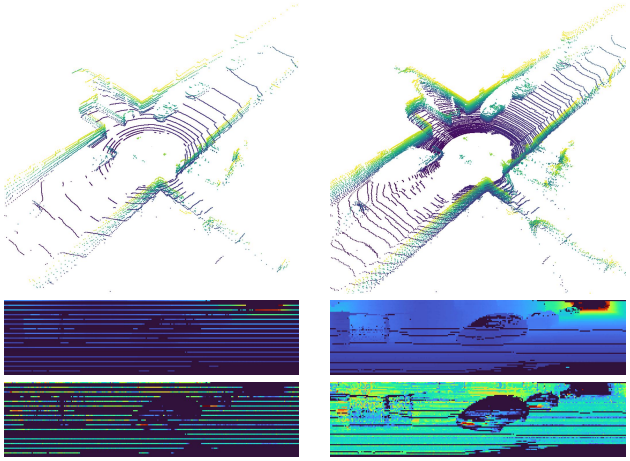


Fig. 1. **LiDAR upsampling using our diffusion model**. We show the sparse LiDAR inputs and our upsampled results as point clouds (top), range images (middle), and reflectance images (bottom). Our results were obtained through image-based conditional generation using our model. The range and reflectance images are partially zoomed for visual purposes.

I. Introduction

LiDAR (light detection and ranging) sensors play a pivotal role in enabling mobile robots to perceive surrounding obstacles and geometry for autonomous navigation. The sensors produce accurate point clouds of 3D scenes by emitting laser beams at omnidirectional angles and detecting reflections from objects. The collected point clouds, along with laser reflectance information, can be utilized for 3D scene understanding tasks [1], such as semantic/instance segmentation and object detection.

However, high-density and high-quality point clouds are not readily accessible on all platforms because increasing the number of beams elevates cost and energy consumption. Therefore, using low-cost yet low-beam LiDAR sensors can be a practical option for system development, while they lead to performance degradation in 3D scene understanding due to the sparsity of the point clouds.

Generative modeling of LiDAR point clouds is a frontier approach to tackle these issues, which aims to learn the prior distribution of 3D scenes. The sparse or incomplete point clouds can be restored by using the learned priors. Motivated by the impressive achievements of deep generative models [2], various approaches have been proposed for LiDAR data generation [3]–[6] using variational autoencoders (VAEs) [7], generative adversarial networks (GANs) [8], and diffusion models [9]–[15].

In this paper, we present R2DM, a novel generative model for LiDAR data that can generate diverse and high-fidelity 3D scene point clouds. As a framework for building a generative model, we employ denoising diffusion probabilistic models (DDPMs) [12], one of the most successful approaches among diffusion models. DDPMs offer various benefits over other frameworks, such as training stability, sample quality, and versatility to inverse problems, as demonstrated in the wide range of domains. We demonstrate the effectiveness of DDPMs on generative modeling of LiDAR data.

Similar to the relevant studies [3]–[6], we cast our task as an image-based generation, wherein LiDAR point clouds are represented by equirectangular images where each pixel contains pointwise range and the corresponding laser reflectance values. To gain the performance of DDPMs on LiDAR data, we first investigate the suitable model design in three aspects: loss functions, data representation, and spatial inductive bias. With our designed architecture, R2DM achieves state-of-the-art generation performance across various levels of metrics, including point clouds, range images, and bird's eye views. Furthermore, the pre-trained R2DM can be applied to LiDAR completion tasks. We propose a flexible pipeline capable of handling various types of corruption, including issues stemming from low-beam LiDAR configurations (see Fig. 1 for instance).

*This work was supported by JSPS KAKENHI Grant Number JP23K16974 and JST [Moonshot R&D] [Grant Number JPMJMS2032]

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

Our contributions can be summarized as follows:

- We present R2DM (range–reflectance diffusion model), a DDPM-based generative model capable of generating diverse and high-fidelity LiDAR range images with a reflectance modality, which demonstrates state-of-the-art performances on the KITTI-360 [16] and KITTI-Raw [17] datasets.
- We provide an in-depth analysis of the effective training of DDPMs in the LiDAR domain. Our key finding is that an explicit spatial bias significantly influences the fidelity of generated samples.
- We also introduce a flexible LiDAR completion pipeline using powerful properties offered by DDPMs, which also outperforms baseline methods on the beam-level upsampling task using KITTI-360.

II. Related Work

A. Generative models

Generative models aim to learn the underlying distribution of a dataset. During the last decade, the rapid advancements in deep neural networks have spurred the development of various frameworks for generative models.

One of the most prominent frameworks is generative adversarial networks (GANs) [8]. A GAN consists of a pair of competing neural networks: a generator and a discriminator, which are trained alternately to minimize the adversarial objective. While GANs are known for their sampling efficiency and high synthesis quality, training instability poses a challenge due to their competitive nature.

Recently, diffusion models, including score-based generative models [9]–[11] and denoising diffusion probabilistic models (DDPM) [12]–[15], have garnered substantial attention, especially in the field of text-to-image generation [15]. Diffusion models have notable advantages that have led to significant advancements. Unlike GANs, they offer stable training with a simple objective function thanks to the formulation approximating likelihood maximization. Training diffusion models is also efficient and scalable since a single neural network suffices to formulate both generative and inference processes. Moreover, the generative process is learned as *iterative refinement*; the task imposed on the model gets easier. Finally, it is worth noting that pre-trained models can be adapted to solving inverse problems, such as colorization, super-resolution, and inpainting [18].

B. LiDAR data generation

Motivated by the progress in the natural image domain, some studies have tackled generative modeling of LiDAR data based on the *range image*-based representation. Caccia *et al.* [3] initiated pivotal work on this subject, utilizing VAEs [7] and GANs [8] to train LiDAR range images. Several studies focused on the discrete dropout noise spread on the range images, called raydrop. Nakashima and Kurazume [4] introduced DUSty, a noise-robust GAN architecture disentangling the noisy range images into denoised image space and the corresponding dropout probability. Building upon the idea of DUSty, Nakashima *et al.* [5] introduced DUSty v2, a neural field-based architecture capable of representing range images at arbitrary resolutions. They also introduced a sim2real application using the learned priors. Despite the steady improvements, these studies [3]–[5] were limited to generating the range modality alone.

LiDARGen proposed by Zyrianov *et al.* [6] is closely related to our work, which is a diffusion model that trains both range and reflectance modalities based on the image representation. In particular, they employed NCSNv2 [10], one of the score-based generative models. NCSNv2 expresses the data distribution $p(\boldsymbol{x})$ by training the gradient of log probability $\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$ involving multi-level noise perturbation, the so-called *score*. Sampling data is performed by Langevin dynamics at stepping-down noise levels. They also proposed to modify the model architecture, incorporating circulating kernel operations and introducing a spatial bias. LiDARGen has demonstrated state-of-the-art results on standard LiDAR datasets, albeit with only subtle improvements compared to existing approaches. As addressed in [6], low sampling efficiency also remains an issue, which is dominated by the number of sampling timesteps determined when training and an evaluation cost per step using the neural network.

The aim of this paper is to improve both the fidelity and efficiency of the diffusion-based approach. Our method R2DM is built upon the DDPM framework with timestep-agnostic training and an efficient neural network architecture. Our experiments reveal that R2DM outperforms LiDARGen with fewer sampling steps and faster network evaluation.

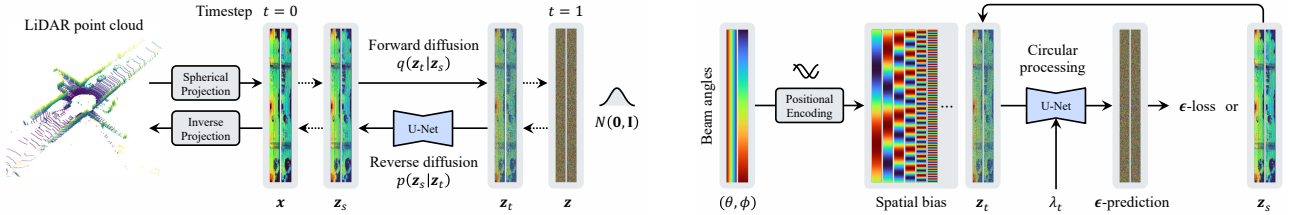
III. Proposed Method

To explore the effective DDPM design for LiDAR data, this section provides the formulation of DDPMs and introduce some modification regarding loss function, data representation, and spatial inductive bias. Lastly, we also introduce the LiDAR completion pipeline using our model.

A. Preliminary

In this paper, we employ the DDPM framework that formulates transitions between data and latent spaces in continuous time $t \in [0, 1]$ [14, 15, 19]. Unlike the discrete-time diffusion models [9, 10, 12] including LiDARGen [6], there is no need to carefully define the number of steps in the training phase, which can be determined in the sampling phase afterward, balancing computational tradeoffs.

In DDPMs, an inference process is called a forward diffusion process, which gradually corrupts the data sample $\boldsymbol{x}$ by adding Gaussian noise as t progresses from 0 to 1, thereby resulting in the latent variable $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. In contrast, a generative process is called a *reverse* diffusion process, which gradually denoise the latent variable $\boldsymbol{z}$ at $t = 1$ to be the data sample $\boldsymbol{x}$ at $t = 0$. The transitional noisy samples $\boldsymbol{z}_t$ for $0 < t < 1$ are also considered as the latent variables. The schematic diagram is depicted in Fig. 2(a).



(a) Range/reflectance image-based diffusion model (b) Details of the reverse diffusion process

Fig. 2. **Overview of R2DM**. (a) The diffusion processes are performed on the range/reflectance image representation. (b) U-Net is trained to recursively denoise the latent variables z_t at $t > 0$, conditioned by the beam angle-based spatial bias and the scheduled signal-to-noise ratio λ_t.

1) Forward diffusion process: Conveniently, since the forward diffusion process follows the additive Gaussian, the noisy samples $\boldsymbol{z}_t$ at arbitrary timestep t can be given by:

$$q(\boldsymbol{z}_t \mid \boldsymbol{x}) = \mathcal{N}(\alpha_t \boldsymbol{x}, \sigma_t^2 \mathbf{I}), \tag{1}$$

where α_t and σ_t are parameters to determine the noising schedule. For example, the most popular schedule is α-cosine schedule [13] where $\alpha_t = \cos(\pi t/2)$ and $\sigma_t = \sin(\pi t/2)$. This transition distribution can be re-parameterized as:

$$\boldsymbol{z}_t = \alpha_t \boldsymbol{x} + \sigma_t \boldsymbol{\epsilon}, \tag{2}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and the signal-to-noise ratio of $\boldsymbol{z}_t$ can be defined as $\lambda_t = \alpha_t^2/\sigma_t^2 = \cot^2(\pi t/2)$. In addition, the transition of latent variables $q(\boldsymbol{z}_t \mid \boldsymbol{z}_s)$ from timestep s to t, for any $0 \leq s < t \leq 1$, can be written as:

$$q(\boldsymbol{z}_t \mid \boldsymbol{z}_s) = \mathcal{N}(\alpha_{t|s} \boldsymbol{z}_s, \sigma_{t|s}^2 \mathbf{I}), \tag{3}$$

where $\alpha_{t|s} = \alpha_t/\alpha_s$ and $\sigma_{t|s}^2 = \sigma_t - \alpha_{t|s}\sigma_s$.

2) Reverse diffusion process: Given the distributions above, the reverse diffusion process $p(\boldsymbol{z}_s \mid \boldsymbol{z}_t)$ is given by:

$$p(\boldsymbol{z}_s \mid \boldsymbol{z}_t) = \mathcal{N}(\boldsymbol{\mu}_t(\boldsymbol{x}, \boldsymbol{z}_t), \Sigma_t^2 \mathbf{I}),$$
$$\boldsymbol{\mu}_t(\boldsymbol{x}, \boldsymbol{z}_t) = \frac{\alpha_{t|s}\sigma_s^2}{\sigma_t^2}\boldsymbol{z}_t + \frac{\alpha_s \sigma_{t|s}^2}{\sigma_t^2}\boldsymbol{x}, \qquad \Sigma_t^2 = \frac{\sigma_{t|s}^2 \sigma_s^2}{\sigma_t^2}. \tag{4}$$

3) Training: The training objective of DDPM is to estimate the unknown $\boldsymbol{x}$ in Eq. 4 by a neural network, where U-Net [20] is generally used. In general, $\boldsymbol{\epsilon}$-prediction and $\boldsymbol{\epsilon}$-loss [12] are preferable; re-parameterizing $\boldsymbol{x}$ as a function of noise $\boldsymbol{\epsilon}$ by Eq. 2. The loss function is given by:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{x}, \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), t \sim \mathcal{U}(0,1)} \left[\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}(\boldsymbol{z}_t, \lambda_t)\|_2^2 \right], \tag{5}$$

where $\hat{\boldsymbol{\epsilon}}(\cdot)$ is the neural network predicting the noise $\boldsymbol{\epsilon}$ from $\boldsymbol{z}_t$ and the corresponding λ_t.

4) Sampling: Once the training is complete, we can sample data by recursively evaluating $p(\boldsymbol{z}_s \mid \boldsymbol{z}_t)$ where $\boldsymbol{x}$ is approximated by $\hat{\boldsymbol{x}} = (\boldsymbol{z}_t - \sigma_t \hat{\boldsymbol{\epsilon}}(\boldsymbol{z}_t, \lambda_t))/\alpha_t$ with a finite number of steps T from $t = 1$ to $t = 0$.

B. Loss function

In Eq. (5), the L_2 loss is used to supervise the model prediction. Meanwhile, in the context of monocular depth estimation, Saxena *et al.* [21] found that an L_1-based formulation offers better performance due to its robustness to large depth values and noisy holes. In our experiment, we also evaluate the L_1 loss and the combination of L_1 and L_2, the Huber loss.

C. Data representation

We assume a LiDAR sensor that has an angular resolution of W in azimuth and H in elevation and measures the range and reflectance at each angle (θ, ϕ). Then, HW sets of the range and reflectance values can be projected to an equirectangular image space $\boldsymbol{x} \in \mathbb{R}^{2 \times H \times W}$ by spherical projection. Here we focus on the range representation and compare three approaches. Similar to LiDARGen [6], we first convert the range $\boldsymbol{d} \in [0, d_{\text{max}}]^{H \times W}$ into the log-scale representation $\boldsymbol{d}_{\log} \in [0, 1]^{H \times W}$ as follows:

$$\boldsymbol{d}_{\log} = \frac{\log(\boldsymbol{d} + 1)}{\log(d_{\text{max}} + 1)}, \tag{6}$$

where d_{max} is the maximum range value. $\boldsymbol{d}_{\log}$ is then re-scaled linearly into $[-1, 1]$ as conventional DDPMs [12]. This log-scale representation gains the resolution of nearby points filling a large portion of the image. We also compare other popular representations of *depth* maps: standard metric depth $\boldsymbol{d}$ and inverse depth (reciprocal) [4, 5], which are also normalized into $[-1, 1]$.

D. Spatial inductive bias

The generated pixels must be accurate and well-aligned with the angular coordinates $\{(\theta, \phi)\}$ to ensure the geometric fidelity in transformed point clouds. Since the initial input $\boldsymbol{z}$ of the neural network $\hat{\boldsymbol{\epsilon}}(\cdot)$ is *i.i.d.* Gaussian noise, an implicit positional encoding by zero padding [22, 23] is considered a mainstay for obtaining structured outputs. LiDARGen [6] proposed concatenating the raw angular coordinates with the input as an explicit spatial inductive bias. However, we hypothesize that the coordinates alone, lacking in high-frequency components and horizontal continuity, are insufficient for representing detailed structures of LiDAR data.

We here generalize the spatial bias as a positional encoding, which can be seen as an identity function in the LiDARGen case. We further investigate *wideband* and *cylindrical* mappings: spherical harmonics [24, 25] and Fourier features [26]. Spherical harmonics represent functions on a sphere through a series of orthogonal functions, which is demonstrated in novel view synthesis [24] and LiDAR compression [25]. We encode the angles by spherical harmonic basis functions. We also compare Fourier features [26]. We employ a $\log_2$-spaced scheme [27], which extends the elevation and azimuth angles separately to a set of power-of-two frequencies so that the mapping preserves the horizontal continuity. Fig. 2(b) depicts the mechanism.

TABLE I

ARCHITECTURE COMPARISON OF DIFFUSION-BASED MODELS

Method	U-Net architecture	# params	msec/step†
LiDARGen [6]	RefineNet [31] in [10]	29,694,082	47.17
R2DM (ours)	Efficient U-Net [15]	31,099,650	**15.77**

† Average time of 1000 runs on our GPU w/ PyTorch compilation.

E. Noise prediction model

Following the standard diffusion models [9]–[15], we use a U-Net architecture [20] to predict the noise $\boldsymbol{\epsilon}$ in Eq. 5. Specifically, our model is built upon Efficient U-Net [15] which was demonstrated on monocular depth estimation [21]. We change the input and output channels as to process the 2-channel LiDAR imagery: range and reflectance. A logarithmic signal-to-noise ratio $\log \lambda_t$ is embedded into affine weights of every adaptive group normalization (AdaGN) [28]. We adopt three residual blocks at each resolution and the multi-head self-attention layer at the lowest resolution. The final model involves 31.1M parameters which was adjusted to roughly align with LiDARGen [6] with 29.7M parameters for fair comparison. Table I shows the architecture comparison. All kernel operations, such as convolution and up/down-resampling, are modified to use circular padding in the horizontal direction, as demonstrated in LiDAR processing tasks [4]–[6, 29, 30].

F. LiDAR completion

Lugmayr *et al.* [18] proposed RePaint, an image inpainting method that exploits the pre-trained unconditional DDPMs. In this work, we build a LiDAR completion pipeline by integrating our R2DM and RePaint. In the sampling phase, $p(\boldsymbol{z}_s \mid \boldsymbol{z}_t)$ in Eq. 4 follows by:

$$\boldsymbol{z}_s = \boldsymbol{m} \odot \boldsymbol{z}_s^{\text{known}} + (1 - \boldsymbol{m}) \odot \boldsymbol{z}_s^{\text{unknown}}, \qquad (7)$$

where $\boldsymbol{m} \in \{0, 1\}^{H \times W}$ is a binary mask indicating known range/reflectance pixels, $\boldsymbol{z}_s^{\text{known}}$ is *noised* known pixels sampled by $q(\boldsymbol{z}_s \mid \boldsymbol{x})$ in Eq. 1, and $\boldsymbol{z}_s^{\text{unknown}}$ is *denoised* unknown pixels sampled by $p(\boldsymbol{z}_s \mid \boldsymbol{z}_t)$ in Eq. 4. Moreover, to harmonize the gap between the known and unknown diffused extents, the sampling step above is repeatedly cycled with the forward diffusion step $q(\boldsymbol{z}_t \mid \boldsymbol{z}_s)$ in Eq. 3. We set the number of sampling steps to 32 and the number of harmonization for each step to 10. Fig. 3 showcases our results on the beam-level, point-level, and object-level simulated corruptions.

IV. EXPERIMENTS

We evaluate our method on a generation task in Section IV-A and a completion task in Section IV-B.

A. Generation task

1) Settings: To evaluate performances on the *unconditional* generation task, we use the KITTI-360 dataset [16]. KITTI-360 consists of 81,106 point clouds measured by Velodyne HDL-64E (64-beam mechanical LiDAR sensor). The data splits are from Zyrianov *et al.* [6]. Each of the point clouds is projected to a 64×1024 image. We compare

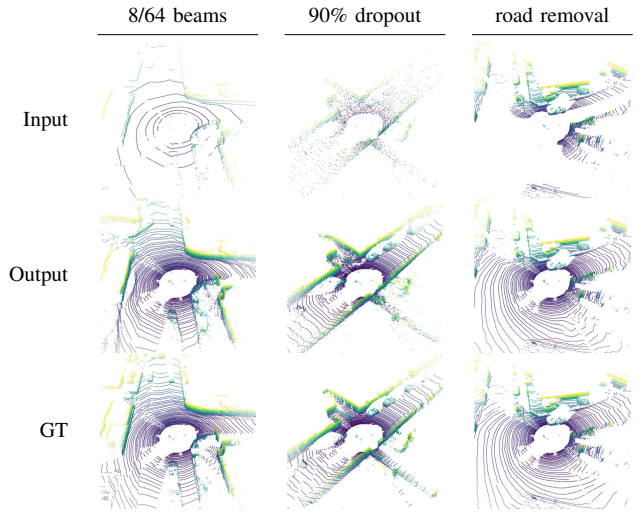


Fig. 3. **Conditional generation using R2DM**. We showcase the simulated corruptions (top) and our restored results (middle). Our method can handle the various levels of completion. The road removal (right) mimics the wet road situation.

different configurations of DDPM in terms of the loss functions, the range representations, and the positional encoding methods, as discussed in Section III. Table II lists a series of our training configurations (config A–H). As the baseline, we compare with the state-of-the-art method, LiDARGen[1] [6]. We also compare the GAN-based methods [3]–[5] using the KITTI-Raw dataset [17]. Following the baselines, we employ the data splits from the Odometry benchmark [17] and the scan unfolding projection [32]. Note that, in this setting, the point cloud is first projected to a full resolution of 64×2048 and then subsampled to 64×512.

2) Implementation details: Our models were trained in 300k steps while calculating an exponential moving average of model weights with a decay rate of 0.995 per every 10 steps. We performed a distributed training with automatic mixed precision (AMP) on two NVIDIA A6000 GPUs. Note that we did not use AMP for LiDARGen to avoid performance degradation. Our model took approximately 20 GPU hours for training, and 30 GPU hours for generating 10k samples in 1024 denoising steps. Our code and pre-trained weights are available at `https://github.com/kazuto1011/r2dm`.

3) Evaluation metrics: For the generation task, we compute three levels of distributional dissimilarity metrics employed in the LiDAR domain [5, 6], between 10k generated samples and all available real samples.

- *Image-based*: Following the baseline [6], we employ Fréchet range distance (FRD) for evaluating range and reflectance images. The images are first fed into the off-the-shelf RangeNet-53 [33] pre-trained a 19-class semantic segmentation task on SemanticKITTI [34]. Then, FRD calculates the Fréchet distance [35] between the generated and real sets on the feature space.

[1]We do not borrow the scores on the proceedings [6] but re-evaluate with the larger samples officially published at `https://github.com/vzyrianov/lidargen`.

TABLE II

QUANTITATIVE COMPARISON OF KITTI-360 GENERATION.

Method (Framework)		NFE	Configurations‡ Loss	Range	Positional encoding	Image FRD ↓	Point cloud FPD ↓	BEV MMD$_{\times 10^4}$ ↓	BEV JSD$_{\times 10^2}$ ↓
LiDARGen (NCSNv2) [6]		1160†	L_2	Log-scale	Identity	579.39	90.29	7.39	7.38
Ours (DDPM)	config A	256	L_2	Log-scale	Identity	202.40	7.11	1.67	4.52
	config B	256	L_1	Log-scale	Identity	382.35	21.42	7.70	8.28
	config C	256	Huber	Log-scale	Identity	174.83	11.20	1.55	4.71
	config D	256	L_2	Metric	Identity	229.28	12.03	1.47	4.01
	config E	256	L_2	Inverse	Identity	188.84	19.66	1.85	3.12
	config F	256	L_2	Log-scale	w/o spatial bias	910.67	253.21	40.45	18.05
	config G	256	L_2	Log-scale	Spherical harmonics	180.60	4.90	2.18	4.12
	config H	256	L_2	Log-scale	Fourier features	**153.73**	**3.92**	**0.68**	**2.17**

† Five steps for each of the 232 noise levels. ‡ The shaded cells indicate the differences from config A.

- *Point cloud-based*: LiDAR range images can be transformed back to 3D point clouds. Following the baseline [5], we also evaluate on the level of point clouds with the Fréchet point cloud distance (FPD) [36]. FPD uses PointNet [37] pre-trained a 16-class classification task on ShapeNet [38] and calculates the Fréchet distance on the feature space like FRD.
- *BEV-based*: Following the baseline [6], we also evaluate on the level of 2D bird's eye view (BEV) [6] projected from the point clouds. We report the Jensen–Shannon divergence (JSD) and maximum-mean discrepancy (MMD), calculating the distance between the marginal point distributions on the 2D BEV histograms.

4) Baseline results: First, we compare LiDARGen [6] and our closest setup (DDPM config A). Fig. 4 shows the four evaluation scores as functions of the number of function evaluations (NFE)[2]. For sampling of our DDPM, we swept T in $\{16, 32, 64, 128, 256, 512, 1024\}$. Intuitively, all scores of the DDPM improve as T increases. Moreover, the DDPM outperformed LiDARGen with significantly lower NFE. Considering the tradeoff between computational time and quality, we will use $T = 256$ for our DDPMs in the following experiments, unless otherwise specified.

5) Ablations: Comparing config A and B in Table II, we can see that L_1 loss is not effective for our settings. We observed that samples of config B is overly smoothed due to the less sensitivity of L_1 loss. Huber loss (config C), the combination of L_1 and L_2, showed the marginal improvements only on FRD and MMD. Regarding the representation of the range modality, the metric depth (config D) and the inverse depth (config E) improved some metrics, while they harmed FPD. Finally, our model with the positional encoding of Fourier features (config H) shows the best results with the large gaps from the other configurations. Without the spatial bias (config F), DDPM degrades all the metrics, even worse than the baseline. We hereinafter call config H as R2DM.

6) Comparison to GANs: Table III compares our R2DM and GANs. When increasing the number of timestep T from the default setting ($T = 256$), R2DM outperforms the baseline in FPD. We believe that the performance gap with the KITTI-360 experiment lies in the setup of range images. In KITTI-360 experiments, the range images were downscaled to alleviate missing points called ray-drop noises. In contrast, the range images of KITTI-Raw were also downscaled but the ray-drop noises were retained to be closer to raw scan data. It is considered that there is room for further ingenuity to handle noisy settings, such as full resolution.

7) Qualitative results: Fig. 5 shows the generated samples of KITTI-360 and KITTI-Raw. We compare the real data, the baseline, and our method. We can see that our method realizes high-fidelity structures of LiDAR point clouds both locally and globally. For instance, regarding the scan lines

[2]NFE represents how many times the neural network is processed in sampling. For LiDARGen (NCSNv2), we count up the number of sampling steps for each noise level, which is 1160 steps in total.

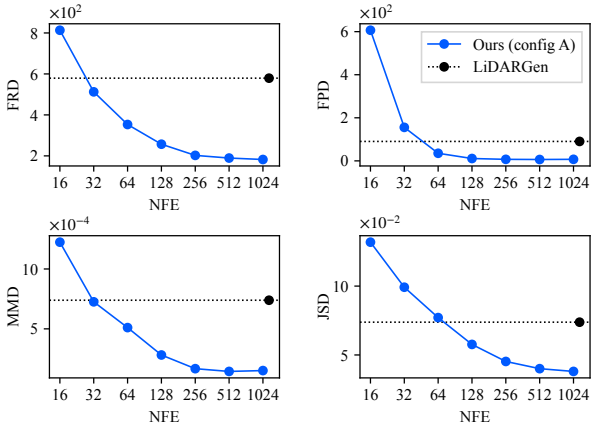


Fig. 4. **Comparison of diffusion-based methods**. For overall metrics, our method achieved better scores with the significantly lower number of function evaluations (NFE), against 1160 steps of LiDARGen [6].

TABLE III

QUANTITATIVE COMPARISON ON KITTI-RAW GENERATION.

Method	Image FRD ↓	Point cloud FPD ↓	BEV MMD$_{\times 10^4}$ ↓	BEV JSD$_{\times 10^2}$ ↓
Vanilla GAN [3, 4]	N/A	3657.60	1.02	5.03
DUSty v1 [4]	N/A	223.63	0.80	2.87
DUSty v2 [5]	N/A	98.02	**0.22**	**2.86**
R2DM ($T = 256$)	215.27	128.74	0.72	3.79
R2DM ($T = 512$)	209.24	89.62	0.65	3.76
R2DM ($T = 1024$)	**207.31**	**70.34**	0.44	3.56

FRD is not available for the baselines [4, 5] which do not support the reflectance.

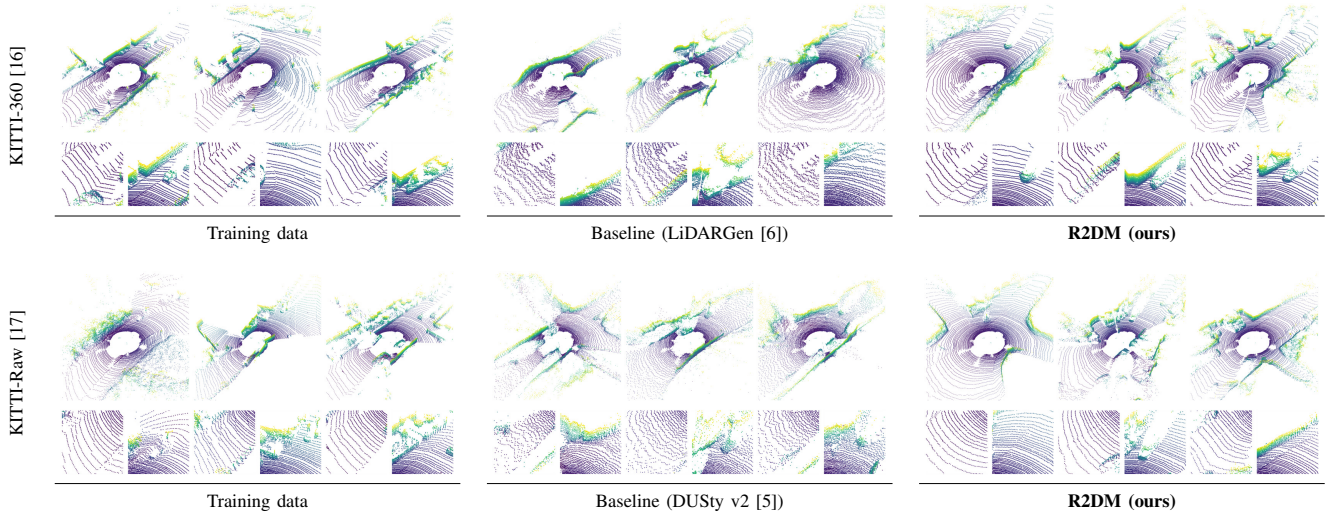


Fig. 5. **Unconditional generation results**. We show the results of the baselines and our method on KITTI-360 (top) and KITTI-Raw (bottom). The LiDARGen results are from officially released samples [6]. The DUSty v2 results are generated using the official pre-trained models [5].

(zoomed regions in Fig. 5), our results are less noisy and close to the real data compared to the baselines.

B. Completion task

1) Settings: Following LiDARGen [6], we conduct a beam-level $4\times$ upsampling experiment on KITTI-360 to assess the completion method described in Section III-F. We simulate the sparse input by subsampling every four horizontal scan lines. We compare our method with simple interpolation methods (nearest-neighbor, bilinear, bicubic), simulation-based supervised methods (LiDAR-SR [39], ILN [40]) trained on the CARLA simulator [41], and the diffusion model-based approach LiDARGen [6]. LiDARGen leverages posterior sampling of score-based generative models, that is, guiding the scores in Langevin dynamics based on the errors of known pixels.

2) Quantitative results: To evaluate point-wise reconstruction performance, we compute mean absolute error (MAE) for both range and reflectance modalities. Furthermore, we evaluate semantic consistency by calculating the intersection-over-union (IoU, %) between the segmentation labels of upsampled and ground truth data. These labels are predicted using the pre-trained RangeNet-53 model [33]. We use ground-truth reflectance as an upper bound for the simulation-based methods, which do not support reflectance modality. The evaluation results are shown in Table IV. Our R2DM outperforms the baselines for all evaluation metrics with large margins.

3) Qualitative results: We provide the results of upsampling and semantic segmentation in Fig. 6. Compared to the baseline, our method shows better fidelity and consistency on the point clouds and the predicted classes.

TABLE IV

BEAM-LEVEL $4\times$ UPSAMPLING ON KITTI-360 TEST SET.

		Range	Reflectance	Semantics
Approach	Method	MAE ↓	MAE ↓	IoU % ↑
Interpolation	Nearest-neighbor	2.083	0.106	18.78
	Bilinear	2.110	0.101	18.17
	Bicubic	2.297	0.108	18.54
Supervised	LiDAR-SR [39]	2.085	N/A	21.61
	ILN [40]	2.237	N/A	21.80
Diffusion	LiDARGen [6]	1.551	0.080	22.46
	R2DM + RePaint [18]	**0.923**	**0.050**	**34.44**

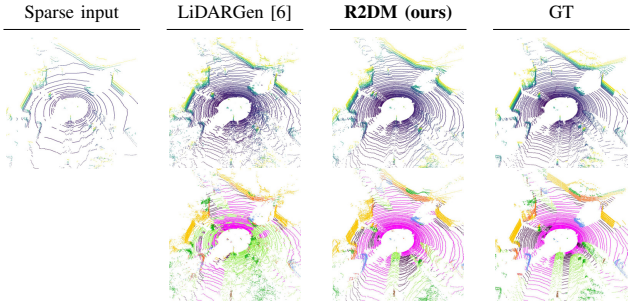


Fig. 6. **Beam-level upsampling results on KITTI-360**. We compare the upsampled point clouds (top) and the semantic segmentation results (bottom) between the diffusion-based methods. The semantic segmentation results are color-coded according to the predicted classes.

V. CONCLUSIONS

We proposed R2DM, a denoising diffusion probabilistic model for realistic LiDAR range/reflectance generation based on the image representation. R2DM achieved the state-of-the-art performance with the lower computational cost against the baseline. Our exploration of the effective model revealed that the fidelity in point clouds can be significantly improved by introducing the explicit spatial bias with Fourier features. Furthermore, we proposed a completion pipeline that leveraged the pre-trained R2DM and demonstrated the beam-level upsampling task. Future work involves investigating model scalability, noise-robust training, and further applications to perception tasks.

REFERENCES

[1] Y. Li, L. Ma, Z. Zhong, F. Liu, M. A. Chapman, D. Cao, and J. Li, "Deep learning for LiDAR point clouds in autonomous driving: A review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 8, pp. 3412–3432, 2020.

[2] S. Bond-Taylor, A. Leach, Y. Long, and C. 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)*, vol. 44, no. 11, pp. 7327–7347, 2022.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

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

[9] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in *Proceedings of the Advances in neural information processing systems (NeurIPS)*, pp. 11895–11907, 2019.

[10] Y. Song and S. Ermon, "Improved techniques for training score-based generative models," in *Proceedings of the Advances in neural information processing systems (NeurIPS)*, vol. 33, pp. 12438–12448, 2020.

[11] 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.

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

[13] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 8162–8171, 2021.

[14] D. Kingma, T. Salimans, B. Poole, and J. Ho, "Variational diffusion models," in *Proceedings of the Advances in neural information processing systems (NeurIPS)*, vol. 34, pp. 21696–21707, 2021.

[15] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, *et al.*, "Photorealistic text-to-image diffusion models with deep language understanding," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, pp. 36479–36494, 2022.

[16] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2D and 3D," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 45, no. 3, pp. 3292–3310, 2022.

[17] 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.

[18] 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.

[19] E. Hoogeboom, J. Heek, and T. Salimans, "Simple diffusion: end-to-end diffusion for high resolution images," in *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 13213–13232, 2023.

[20] 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.

[21] S. Saxena, A. Kar, M. Norouzi, and D. J. Fleet, "Monocular depth estimation using diffusion models," *arXiv:2302.14816*, 2023.

[22] R. Xu, X. Wang, K. Chen, B. Zhou, and C. C. Loy, "Positional encoding as spatial inductive bias in gans," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13569–13578, 2021.

[23] J. Choi, J. Lee, Y. Jeong, and S. Yoon, "Toward spatially unbiased generative models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14253–14262, 2021.

[24] D. Verbin, P. Hedman, B. Mildenhall, T. Zickler, J. T. Barron, and P. P. Srinivasan, "Ref-NeRF: Structured view-dependent appearance for neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5481–5490, 2022.

[25] K. Zhang, Z. Hong, S. Xu, and S. Wang, "CURL: Continuous, ultra-compact representation for LiDAR," in *Proceedings of the Robotics: Science and Systems (RSS)*, 2022.

[26] M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron, and R. Ng, "Fourier features let networks learn high frequency functions in low dimensional domains," in *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 7537–7547, 2020.

[27] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 405–421, 2020.

[28] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," in *Proceedings of the Advances in neural information processing systems (NeurIPS)*, vol. 34, pp. 8780–8794, 2021.

[29] K. Nakashima, H. Jung, Y. Oto, Y. Iwashita, R. Kurazume, and O. M. Mozos, "Learning geometric and photometric features from panoramic LiDAR scans for outdoor place categorization," *Advanced Robotics*, vol. 32, no. 14, pp. 750–765, 2018.

[30] S. Schubert, P. Neubert, J. Pöschmann, and P. Protzel, "Circular convolutional neural networks for panoramic images and laser data," in *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, pp. 653–660, 2019.

[31] G. Lin, A. Milan, C. Shen, and I. Reid, "Refinenet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1925–1934, 2017.

[32] L. T. Triess, D. Peter, C. B. Rist, and J. M. Zöllner, "Scan-based semantic segmentation of LiDAR point clouds: An experimental study," in *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, pp. 1116–1121, 2020.

[33] 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.

[34] 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.

[35] D. C. Dowson and B. V. Landau, "The Fréchet distance between multivariate normal distributions," *Journal of Multivariate Analysis*, vol. 12, no. 3, pp. 450–455, 1982.

[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] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 652–660, 2017.

[38] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An information-rich 3D model repository," *arXiv:1512.03012*, 2015.

[39] T. Shan, J. Wang, F. Chen, P. Szenher, and B. Englot, "Simulation-

based lidar super-resolution for ground vehicles," *Robotics and Autonomous Systems (RAS)*, vol. 134, p. 103647, 2020.

[40] Y. Kwon, M. Sung, and S.-E. Yoon, "Implicit LiDAR network: Lidar super-resolution via interpolation weight prediction," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pp. 8424–8430, 2022.

[41] 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.