Scaling Diffusion Models to Real-World 3D LiDAR Scene Completion

Lucas Nunes[1] Rodrigo Marcuzzi[1] Benedikt Mersch[1]
Jens Behley[1] Cyrill Stachniss[1,2]
[1]Center for Robotics, University of Bonn [2]Lamarr Institute for Machine Learning and Artificial Intelligence
{firstname.lastname}@igg.uni-bonn.de

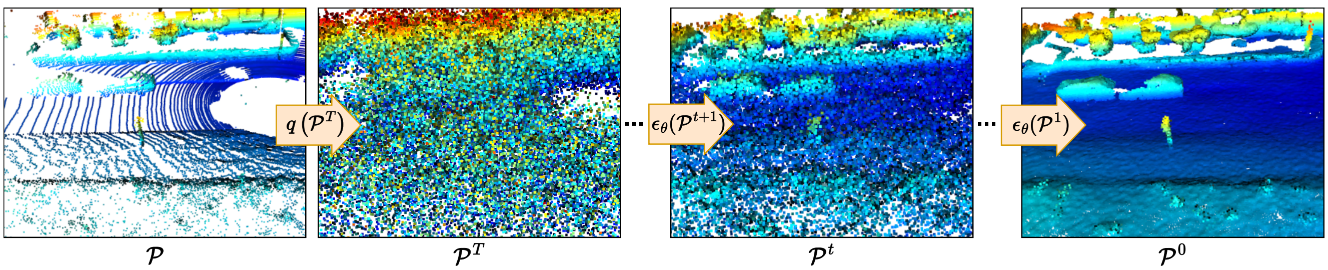

Figure 1. Starting from a single input scan $\mathcal{P}$, we add Gaussian noise to each point, defining the noisy input $\mathcal{P}^T$. Then, we use our trained noise predictor ϵ_θ to denoise $\mathcal{P}^T$ iteratively until arriving at $\mathcal{P}^0$, yielding a completed representation of the 3D scene.

Abstract

Computer vision techniques play a central role in the perception stack of autonomous vehicles. Such methods are employed to perceive the vehicle surroundings given sensor data. 3D LiDAR sensors are commonly used to collect sparse 3D point clouds from the scene. However, compared to human perception, such systems struggle to deduce the unseen parts of the scene given those sparse point clouds. In this matter, the scene completion task aims at predicting the gaps in the LiDAR measurements to achieve a more complete scene representation. Given the promising results of recent diffusion models as generative models for images, we propose extending them to achieve scene completion from a single 3D LiDAR scan. Previous works used diffusion models over range images extracted from LiDAR data, directly applying image-based diffusion methods. Distinctly, we propose to directly operate on the points, reformulating the noising and denoising diffusion process such that it can efficiently work at scene scale. Together with our approach, we propose a regularization loss to stabilize the noise predicted during the denoising process. Our experimental evaluation shows that our method can complete the scene given a single LiDAR scan as input, producing a scene with more details compared to state-of-the-art scene completion methods. We believe that our proposed diffusion process formulation can support further research in diffusion models applied to scene-scale point cloud data. [1]

[1]Code: https://github.com/PRBonn/LiDiff

1. Introduction

Perception systems are a crucial component of self-driving cars, enabling them to understand their surroundings and safely navigate through it. Such systems rely on the data collected by the sensors installed on the vehicle to perceive the environment but fail to deduce areas only partially observable by the sensor. For a human it is comparably rather simple to infer the complete scene from the scene context. Especially in autonomous driving, LiDAR sensors are employed to collect 3D information of the vehicle surroundings to enable safe navigation. Despite the accuracy of those sensors, collected point clouds are sparse, with large gaps between the data points measured by the sensor beams. Being able to complete the measured scene can add valuable information to perception systems, helping to improve different tasks such as object detection [42], localization [40] or navigation [29].

Scene completion tries to infer the missing parts of a scene, providing a dense and more complete scene representation. Given the LiDAR data sparsity, having a way to fill the gaps of non-observed regions is helpful to enlarge the incomplete data measured by the sensor. Previously, this task was tackled using paired RGB images and LiDAR point clouds by inferring depth maps from an RGB image supervised by the LiDAR depth measurements [8, 21, 22, 44]. Other approaches [15, 25, 40] employ signed distance fields (SDF) where the scene is represented as a voxel grid where each voxel stores its distance to the closest surface in the point cloud. Such methods approximate the scene by a surface representation, losing details usually present in real-world data since these approaches are limited to the voxel resolution. As an extension to this task, semantic scene completion has emerged [15, 31, 32], where the goal is to infer an occupancy voxel grid with a semantic label associated to each voxel. However, those methods require large amounts of labeled data and operate at a predefined fixed voxel grid resolution. More recently, denoising diffusion probabilistic models (DDPM) were employed in the context of self-driving cars [14, 26, 50] relying on image representations of the LiDAR data, such as range images [26, 50] or a discrete diffusion process formulation, inferring the occupancy on a predefined voxel grid [14].

In this work, we propose a diffusion scheme for 3D data operating at point level and at scene scale. We exploit the generative properties of DDPMs to infer the unseen regions of a scene measured by a 3D LiDAR sensor, achieving scene completion from a single point cloud as illustrated in Fig. 1. We reformulate the (de)noising scheme used in DDPMs by adding noise locally to each point without scaling the input data to the noise range, allowing the model to learn detailed structural information of the scene. Furthermore, we propose a regularization to stabilize the DDPMs during training, approximating the predicted noise distribution closer to the real data. We compare our method with different scene completion approaches and conduct extensive experiments to validate our proposed scene-scale 3D diffusion scheme. In summary, our key contributions are:

- We propose a novel scene-scale diffusion scheme for 3D sensor data that operates at the point level.
- We propose a regularization that approximates the predicted noise to the expected noise distribution.
- Our method can generate more fine-grained details compared to previous methods.
- Our approach achieves competitive performance in scene completion compared to previous diffusion and non-diffusion methods.

2. Related Work

Scene completion aims at inferring missing 3D scene information given an incomplete sensor measurement. This inference of unseen information can be helpful for perception tasks [42], localization [40] or navigation [29]. Some works [8, 21, 22, 44] tackled this task by jointly extracting information from paired RBG images and LiDAR point clouds, predicting a depth map from an RGB image supervised by the LiDAR data. Differently, other methods [15, 40] approach the problem by optimizing a signed distance field (SDF) given only the LiDAR measurements, representing the scene as a voxel grid where each voxel stores its distance to the closest surface in the scene. However, such methods are bound to the voxel resolution and lose details in the scene due to the discretization by voxels. Distinctly, our approach works directly on the points and exploits the generative properties of DDPMs to complete the unseen data without relying on a voxel grid representation.

Semantic scene completion has been of great interest more recently due to the availability of large datasets with semantic labels [2–4, 7, 9, 16, 39]. This task extends the scene completion task by predicting a semantic label for each occupied voxel [15, 31, 32]. However, those methods are also tightly bound to the voxel grid resolution, which usually has a low resolution due to memory limitations. Besides operating at point level, given the recent research effort for DDPMs, our method could also later be extended to predict a semantic class for each generated point.

Denoising diffusion probabilistic models have gained attention due to their high-quality results in image generation [6, 11, 27, 28, 30, 33, 48, 49]. Besides that, conditioned diffusion models gained even more relevance due to the possibility of generating data towards an input condition [1, 10, 46]. The drawback of DDPMs is usually the time needed during the denoising process. For that reason, many efforts have been put to achieve a faster generation, *e.g*., by doing a distillation of the denoising model [23, 34] or by analytically approximating the denoising steps solution to reduce the amount of steps needed [12, 17, 18, 37].

Diffusion models for 3D data have been investigated due to their promising performance in the image domain. Such methods [19, 20, 35, 36, 43, 45, 47] are focused on single object shapes, achieving novel object shape generation or completion. Few works [14, 26, 50] target real-world data generation. Some works [26, 50] rely on projecting the 3D data to an image-based representation such as range images, such that the methods proposed in the image domain can be directly applied. For such approaches, the 3D scene cannot be completed since when reprojecting the image to the 3D world, some regions do not have any information due to occlusions in the projected point cloud. Lee *et al*. [14] achieves scene-scale 3D data generation using a discrete diffusion model formulation and a fixed voxel grid representation of the environment. The model is then used to infer for each voxel whether it is occupied, and a semantic label is predicted. Different from previous works, our method operates directly at point level and does not rely on a grid representation or projection to the image domain.

Given the recent advances in DDPMs for data generation, we propose a formulation of the denoising diffusion process that works at point level, achieving competitive performance in scene-scale diffusion scene completion. Our formulation enables the use of DDPMs to generate scene scale, real-world-like data without relying on any discretization or projection of the LiDAR data.

3. Approach

We propose using DDPMs to achieve scene completion from a single 3D LiDAR scan as input. First, we reformulate the DDPMs [19, 20, 47] to work at scene scale. Instead of normalizing the input point cloud, we add and predict the noise locally for each point. During the denoising process, we condition the noise prediction with the input scan such that the final scene retains the structural information from the input scan while inferring the missing parts. In this formulation, the initial point cloud is a noisy version of the input scan and the networks task is to denoise it to get the complete scene as depicted in Fig. 1. Next, we provide the needed background on diffusion models and describe the individual components of our approach.

3.1. Denoising diffusion probabilistic models

Denoising diffusion probabilistic models [6, 11, 27] formulate the data generation as an iterative denoising process. Commonly, the model starts from Gaussian noise [6, 11, 27] and iteratively removes noise from the input until it converges to the target output (*e.g.*, images [6, 11, 27, 28, 30, 33, 48, 49] or shapes [19, 20, 35, 36, 43, 45, 47]). This can be achieved by defining a forward diffusion process where noise is iteratively added T times to the target data. Then, the model is trained to predict the noise added at each step t. By predicting the noise at each step t and removing it, the denoised sample should be closer to the target training data.

The diffusion process as formulated by Ho *et al.* [11] can be generally written as follows. Given a sample $\boldsymbol{x}^0 \sim q(\boldsymbol{x})$ from a target data distribution, the diffusion process adds noise to $\boldsymbol{x}^0$ over T steps, resulting in $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^T$, where $q(\boldsymbol{x}^T) \approx \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, where $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ is a normal distribution with mean $\mathbf{0}$ and the identity matrix $\boldsymbol{I}$ as diagonal covariance. This diffusion process is parameterized by a sequence of defined noise factors $\beta_1, \ldots, \beta_T$, where iteratively at each step t, Gaussian noise is sampled and added to $\boldsymbol{x}^{t-1}$ given β_t. This can be simplified to sample $\boldsymbol{x}^t$ from $\boldsymbol{x}^0$, without computing the intermediary steps $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^{t-1}$. To do so, Ho *et al.* [11] define $\alpha_t = 1 - \beta_t$ and $\overline{\alpha}_t = \prod_{i=1}^{t} \alpha_i$, and $\boldsymbol{x}^t$ can be sampled as:

$$\boldsymbol{x}^t = \sqrt{\overline{\alpha}_t}\boldsymbol{x}^0 + \sqrt{1 - \overline{\alpha}_t}\boldsymbol{\epsilon}, \tag{1}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Note that when T is large enough $q(\boldsymbol{x}^T) \approx \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, since $\overline{\alpha}_T$ gets closer to zero.

The denoising process aims to undo the T noising steps by predicting the noise $\boldsymbol{\epsilon}$ added at each step t [11]. Given an initial $\boldsymbol{x}^T$, we want to reverse the diffusion process and get to $\boldsymbol{x}^0$. The reverse diffusion step can be written as:

$$\boldsymbol{x}^{t-1} = \boldsymbol{x}^t - \frac{1 - \alpha_t}{\sqrt{1 - \overline{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}^t, t) + \frac{1 - \overline{\alpha}_{t-1}}{1 - \overline{\alpha}_t}\beta_t \mathcal{N}(\mathbf{0}, \boldsymbol{I}), \tag{2}$$

where $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}^t, t)$ is the noise predicted from $\boldsymbol{x}^t$ at step t.

This generation can also be guided given a condition $\boldsymbol{c}$. This conditional generation can either stem from a pre-trained encoder [6] or from classifier-free guidance [10], where the encoder is trained together with the noise predictor. In our case, we use the classifier-free guidance since it does not require a pre-trained encoder. With the classifier-free guidance, the model is trained to learn the conditional and unconditional noise distribution. In this case, at each training step the model has a probability p of predicting the unconditional noise distribution, where the conditioning is set to a null token, *i.e.*, $\boldsymbol{c} = \emptyset$.

The training process optimizes the denoising model to predict the noise $\boldsymbol{\epsilon}$ added at step t to a given input. Given an input $\boldsymbol{x}^0$ and a condition $\boldsymbol{c}$, a random step $t \in [0, T]$ is sampled, and $\boldsymbol{x}^t$ is sampled from Eq. (1) with a Gaussian noise $\boldsymbol{\epsilon}$. Then, from $\boldsymbol{x}^t$, $\boldsymbol{c}$ and t, the model computes the noise prediction, supervising it with an $\mathcal{L}_2$ loss:

$$\mathcal{L}(\boldsymbol{x}^t, \tilde{\boldsymbol{c}}, t) = \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}^t, \tilde{\boldsymbol{c}}, t) \right\|^2, \tag{3}$$

with $\tilde{\boldsymbol{c}} \sim \mathcal{B}(p)$ where $\mathcal{B}$ is a Bernoulli distribution with outcomes $\{\emptyset, \boldsymbol{c}\}$ with probability p that $\emptyset$ occurs.

The inference starts from an initial $\boldsymbol{x}^T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and iteratively denoise it to get $\boldsymbol{x}^0$. For the classifier-free guidance [10], we predict the conditional and unconditional noise distribution and compute the final predicted noise as:

$$\boldsymbol{\epsilon}'_\theta(\boldsymbol{x}^t, \boldsymbol{c}, t) = \boldsymbol{\epsilon}_\theta(\boldsymbol{x}^t, \emptyset, t) + s\left[\boldsymbol{\epsilon}_\theta(\boldsymbol{x}^t, \boldsymbol{c}, t) - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}^t, \emptyset, t)\right], \tag{4}$$

where $s \in \mathbb{R}$ is a parameter that weights the conditioning to $\boldsymbol{c}$, and $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}^t, \emptyset, t)$ is the unconditional noise prediction.

With Eq. (4) we can compute the noise at any step t, from which we can use Eq. (2) to compute $\boldsymbol{x}^{T-1}, \ldots, \boldsymbol{x}^0$, where $\boldsymbol{x}^0$ is a newly generated sample conditioned on $\boldsymbol{c}$.

3.2. Diffusion scene completion

In this work, we use the generative aspect of DDPMs to complete a scene measured in a single scan by a LiDAR sensor. Similarly to shape completion [19, 20, 47], the input is a partial point cloud $\mathcal{P} = \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_N\}$ where $\boldsymbol{p} \in \mathbb{R}^3$, and the output should be the complete point cloud $\mathcal{P}' = \{\boldsymbol{p}'_1, \ldots, \boldsymbol{p}'_M\}$ where $\boldsymbol{p}' \in \mathbb{R}^3$. In our case, the partial point cloud is a single LiDAR scan from which we want to achieve scene completion. Given a sequence of consecutive LiDAR scans and their poses, we can build a map and sample the complete scene ground truth $\mathcal{G}$ for an individual scan $\mathcal{P}$, where our scene completion $\mathcal{P}'$ should be as close as possible to $\mathcal{G}$.

Given the pair of input scan $\mathcal{P}$ and ground truth $\mathcal{G}$, we can train the DDPM to achieve scene completion. As detailed in Sec. 3.1, we can compute a noisy point cloud $\mathcal{G}^t$ at step t from the complete scene $\mathcal{G}$ in a point-wise fashion:

$$\boldsymbol{p}^t_m = \sqrt{\overline{\alpha}_t}\boldsymbol{p}_m + \sqrt{1 - \overline{\alpha}_t}\epsilon, \ \forall \boldsymbol{p}_m \in \mathcal{G}, \tag{5}$$

with $\mathcal{G}^t = \{\boldsymbol{p}^t_1, \ldots, \boldsymbol{p}^t_M\}$.

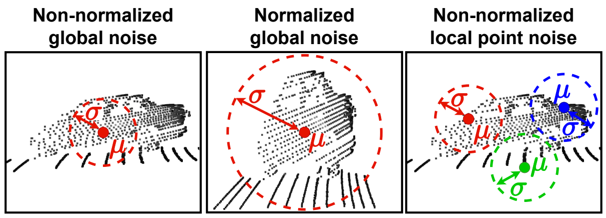

Figure 2. Comparison between Gaussian noise with standard deviation σ and mean μ over non-normalized and normalized input point cloud and our proposed local point-wise noise formulation.

In our case, we want to retrieve the complete scene $\mathcal{G}$ from $\mathcal{G}^T$. However, $\mathcal{G}^T$ retains little information from $\mathcal{G}$ due to the T diffusion steps. Therefore, we condition the generation with the scan $\mathcal{P}$ such that its structure guides the point cloud generation. From Eq. (4), the point-wise classifier-free noise prediction at step t can be rewritten as:

$$\epsilon'_\theta(\mathcal{G}^t, \mathcal{P}, t) = \epsilon_\theta(\mathcal{G}^t, \emptyset, t) + s\big[\epsilon_\theta(\mathcal{G}^t, \mathcal{P}, t) - \epsilon_\theta(\mathcal{G}^t, \emptyset, t)\big]. \quad (6)$$

For training, at each iteration we select a random step $t \in [0, T]$ and compute $\mathcal{G}^t$ from $\mathcal{G}$ given Gaussian noise $\epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Then, we use the model to predict the noise from $\mathcal{G}^t$ conditioned to the LiDAR scan $\mathcal{P}$ or a null token $\emptyset$ given a probability p as in Eq. (3), supervising with the loss:

$$\mathcal{L}_{\text{diff}}(\mathcal{G}^t, \tilde{\boldsymbol{c}}, t) = \big\|\epsilon - \epsilon_\theta(\mathcal{G}^t, \tilde{\boldsymbol{c}}, t)\big\|^2, \quad (7)$$

where as in Eq. (3), $\tilde{\boldsymbol{c}} \sim \mathcal{B}(p)$ with $\mathcal{B}$ as a Bernoulli distribution with outcomes $\{\emptyset, \mathcal{P}\}$ with probability p that $\emptyset$ occurs.

During inference, as detailed in Sec. 3.1, we can generate a scene conditioned to a LiDAR scan $\mathcal{P}$, by denoising from $\mathcal{G}^T$ to $\mathcal{G}^0$ which is the predicted completion $\mathcal{P}'$.

3.3. Local point denoising

The formulation detailed in Sec. 3.2 is usually used for shape completion [20, 47]. Even though achieving promising results for shape completion, this formulation may not directly work at the scene scale. For single object shapes, the data is either normalized or within a small range close to a Gaussian distribution with mean $\mu = \mathbf{0}$ and standard deviation $\Sigma = \boldsymbol{I}$. For scene scale, the LiDAR data has a much larger scale, and the data range differs depending on the point cloud axis. Therefore, the input data distribution is far from a Gaussian distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$, and if we normalize the data, we lose many details in the scene due to compressing it into a much smaller range as illustrated in Fig. 2.

To overcome this problem, we reformulate the diffusion process as a point-wise local problem. Instead of sampling $\boldsymbol{x}^t$ as a mixed distribution between $\epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and $\boldsymbol{x}^0$ as in Eq. (1), we formulate the diffusion process as a noise offset added locally to each point $\boldsymbol{p}_m \in \mathcal{G}$. In this case, from Eq. (1), we set $\boldsymbol{x}^0 = \mathbf{0}$ and add $\boldsymbol{x}^t$ to $\boldsymbol{p}_m$:

$$\boldsymbol{p}_m^t = \boldsymbol{p}_m + \left(\sqrt{\overline{\alpha}_t}\mathbf{0} + \sqrt{1 - \overline{\alpha}_t}\epsilon\right), \quad (8)$$
$$= \boldsymbol{p}_m + \sqrt{1 - \overline{\alpha}_t}\epsilon. \quad (9)$$

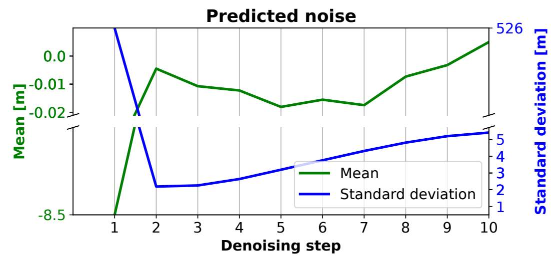

Figure 3. Mean and standard deviation of the predicted noise ϵ_θ without the noise regularization. In this experiment we use DPM-Solver [17] to reduce the denoising steps from $1,000$ to 10.

With this formulation, the noise ϵ is a random offset scaled w.r.t. the step t added to each point $\boldsymbol{p}_m$ in $\mathcal{G}$. The model needs to predict the noise at each step t, slowly moving the noisy points towards the target scene $\mathcal{G}$ conditioned to the LiDAR scan $\mathcal{P}$, still operating in the original scale.

During inference, due to this local diffusion formulation, $\mathcal{G}^T$ cannot be approximated by a Gaussian distribution. Instead, we can generate $\mathcal{G}^T$ from the LiDAR scan $\mathcal{P}$. Besides, to complete the LiDAR scan, we need more points than the input scan. Therefore, given a single LiDAR scan $\mathcal{P}$, we increase its size by concatenating its points K times to get $\mathcal{P}^* = \{\boldsymbol{p}_1^*, \ldots, \boldsymbol{p}_{KN}^*\}$, where $M = KN$. Then, we sample a Gaussian noise for each point $\boldsymbol{p}_m^* \in \mathcal{P}^*$ and compute the initial noisy point cloud $\mathcal{P}^T$ from $\mathcal{P}^*$ with Eq. (9). Finally, we calculate the T denoising steps by predicting the noise at step t from Eq. (4), and denoising it with Eq. (2) to get the complete scan $\mathcal{P}' = \mathcal{P}^0$.

Note that, as long as $\mathcal{P}^T$ is "noisy enough" to resemble $\mathcal{G}^T$ as seen during training, the generation process is the same independent of using $\mathcal{P}^*$ or the ground truth $\mathcal{G}$ to sample the initial $\boldsymbol{x}^T$.

3.4. Noise prediction regularization

DDPMs use a leveraged formulation to train the model to predict only the noise added to the data. This formulation has only to optimize an $\mathcal{L}_2$ loss between the added noise and the model prediction. However, this formulation optimizes the model to precisely predict the noise added to each point, ignoring the overall distribution of the noise sampled.

Given that the added noise $\epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, it is reasonable to expect that the prediction $\epsilon_\theta(\mathcal{G}^t, \mathcal{P}, t) \approx \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. However, the model predicts a peaky distribution far from the expected, as shown in Fig. 3. The predicted noise starts with a mean far from zero and with a large standard deviation. As the denoising starts the mean gets closer to zero but the standard deviation is still far from one. Therefore, we propose a regularization to approximate $\epsilon_\theta(\mathcal{G}^t, \mathcal{P}, t)$ to $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$. We compute the mean $\overline{\epsilon}_\theta$ and the standard deviation $\hat{\epsilon}_\theta$ over $\epsilon_\theta(\mathcal{G}^t, \mathcal{P}, t)$ and calculate the regularization losses:

$$\mathcal{L}_{\text{mean}} = \overline{\epsilon}_\theta{}^2 \quad \text{and} \quad \mathcal{L}_{\text{std}} = (\hat{\epsilon}_\theta - 1)^2, \quad (10)$$

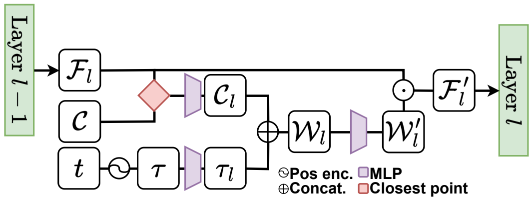

Figure 4. Diagram of the conditioning in each layer l.

then our final loss becomes:

$$\mathcal{L} = \mathcal{L}_{\text{diff}} + r\left(\mathcal{L}_{\text{mean}} + \mathcal{L}_{\text{std}}\right), \tag{11}$$

where r is a weighting factor.

With this regularization, we aim at smoothing the peaky distribution from the predicted noise, and approximating it to the expected distribution, *i.e.*, $\boldsymbol{\epsilon}_\theta(\mathcal{G}^t, \mathcal{P}, t) \approx \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

3.5. Refinement network

Despite the impressive results from DDPMs, the denoising process still demands time since it has to predict all the T steps. Recent efforts [12, 17, 18, 23, 34, 37] focus on increasing the inference speed. However, by reducing the inference time, the generation quality may also decrease. Besides, processing 3D scene-scale data demands many computational resources. This limitation hinders the number of points we can generate to represent the complete scene. Therefore, we follow the refinement upsampling scheme proposed by Lyu *et al.* [20]. As done by them, we train an additional model to refine the scene generated by the diffusion process while upsampling it by predicting κ offsets $\boldsymbol{o}_\kappa \in \mathbb{R}^3$ for each point in the completed scene $\mathcal{P}'$. Then, we add the offsets to the completed scene points as $\{\boldsymbol{p}'_n + \boldsymbol{o}_0, \ldots, \boldsymbol{p}'_n + \boldsymbol{o}_\kappa\}$, $\forall\, \boldsymbol{p}'_n \in \mathcal{P}'$ refining the diffusion completion while upsampling it.

3.6. Noise predictor architecture

We parameterize the denoising process using a MinkUNet [5] as the noise predictor which uses sparse convolutions to process 3D data. To encode information from the conditioning scan $\mathcal{P}$, we use the encoder part from MinkUNet with the same architecture as the noise predictor. The encoder downsamples $\mathcal{P}$ to a smaller version $\mathcal{C} = \{\boldsymbol{c}_{n'} \in \mathbb{R}^{d_c} \mid 1 < n' < N'\}$, where $N' < N$ and d_c is the encoder output embedding size. To encode the denoising step t, similar to previous work [47], we use a sinusoidal positional encoding to compute the temporal embedding $\tau \in \mathbb{R}^{d_t}$. Then, before each layer l in the denoising model, we compute the closest point between the layer input points $\mathcal{F}_l = \{\boldsymbol{f}_{n'_l} \in \mathbb{R}^{d_l} \mid 1 < n'_l < N'_l\}$ and the conditioning embeddings $\mathcal{C}$ to get a per-point guidance, passing it over an MLP to get $\mathcal{C}_l = \{\boldsymbol{c}_{n'_l} \in \mathbb{R}^{d_l} \mid 1 < n'_l < N'_l\}$. Then, we compute $\tau_l \in \mathbb{R}^{d_l}$ from τ through another MLP, and concatenate τ_l to each point in $\mathcal{C}_l$ to obtain $\mathcal{W}_l = \{\boldsymbol{w}_{n'_l} \in \mathbb{R}^{2d_l} \mid 1 < n'_l < N'_l\}$. Finally, we use one more MLP layer to project $\mathcal{W}_l$ to the layer feature dimension d_l and get $\mathcal{W}'_l$. Then, we compute $\mathcal{F}'_l = \mathcal{W}'_l \odot \mathcal{F}_l$ as an element-wise multiplication, which is then feed as the input to layer l, as depicted in Fig. 4. As the refinement network, we use the same MinkUNet architecture used for the noise predictor without the conditioning encoder. For more details on the embeddings dimensions, noise predictor and refinement network architectures, we refer to the supplementary material.

4. Experiments

Datasets. For training our DDPM, we used the SemanticKITTI dataset [2, 9], an autonomous driving benchmark with point-wise annotations over sequences of LiDAR scans collected in an urban environment. To generate the ground truth complete scans, we used the dataset poses to aggregate the scans in the sequence and remove moving objects with the semantic labels, building a map for each sequence. For evaluation, we used the validation set from SemanticKITTI, *i.e.*, sequence 08. Additionally, we used sequence 00 from the KITTI-360 dataset [16] and collected our own data with an Ouster LiDAR OS-1 with 128 beams to further compare the approaches.

For SemanticKITTI and KITTI-360, we used the ground truth poses to build the map, and for our data, we used KISS-ICP [41] to get the scan poses for our sequence. To remove the moving objects from the map in KITTI-360 and our data, we used an off-the-shelf moving object segmentation [24]. To compute the evaluation metrics, for each scan in the sequences, we remove the moving objects using the semantic labels using only the static points as input to the scene completion methods. Then, we evaluate the completed scene by comparing it with the corresponding region in the ground truth map.

Training. We train our model for 20 epochs, using only the training set from SemanticKITTI. As optimizer, we used Adam [13] with a learning rate of 10^{-4} decreased by half every 5 epochs, and decay of 10^{-4}, with batch size equal to 2. For the diffusion parameters, we used $\beta_0 = 3.5 \cdot 10^{-5}$ and $\beta_T = 0.007$, with the number of diffusion steps $T = 1000$, linearly interpolating between β_0 and β_T to define $\beta_1, \ldots, \beta_{T-1}$. We set the noise regularization $r = 5.0$, and the classifier-free probability $p = 0.1$. For the MinkUNet parameters, we set the quantization resolution to $0.05\,\text{m}$. For each input scan, we define the scan range as $50\,\text{m}$ and sample $18,000$ points with farthest point sampling. For the ground truth, we randomly sample $180,000$ points without replacement. For the refinement network we use $\kappa = 6$ as the number of offsets.

Inference. During inference, we use DPMSolver proposed by Lu *et al.* [17], reducing the number of denoising

Method	CD [m] ↓	JSD_{BEV} [m] ↓
LMSCNet [32]	0.641	0.431
LODE [15]	1.029	0.451
MID [40]	0.503	0.470
PVD [47]	1.256	0.498
Ours	**0.375**	**0.416**

Table 1. Mean chamfer distance and Jensen-Shannon divergence evaluation on validation set from SemanticKITTI.

	KITTI-360		Our data	
Method	CD [m] ↓	JSD_{BEV} [m] ↓	CD [m] ↓	JSD_{BEV} [m] ↓
LMSCNet [32]	0.979	0.496	0.826	0.439
LODE [15]	1.565	0.483	**0.387**	0.389
MID [40]	0.637	0.476	0.475	0.379
Ours	0.564	0.459	0.518	0.360
Ours refined	**0.517**	**0.446**	0.471	**0.341**

Table 2. Mean chamfer distance and Jensen-Shannon divergence evaluation on KITTI-360 sequence 00 and our data.

steps T from $1,000$ to 50. Besides, we set the classifier-free conditioning weight to $s = 6.0$. To maintain the same amount of points used during training, we again use the scan max range as $50\,\text{m}$ and sample $18,000$ points with farthest point sampling. Furthermore, as explained in Sec. 3.3, we set $K = 10$ to define the input noisy scan $\mathcal{P}^*$.

Baselines. We compare our method with different scene completion methods, LMSCNet [32], PVD [47], Make It Dense (MID) [40], and LODE [15]. For all baselines, we used their official code and the provided weights also trained on SemanticKITTI. For PVD, we trained the approach with SemanticKITTI with their default parameters. We also follow their data loading, where the point clouds are normalized before the diffusion process. LMSCNet [32] and LODE [15] are limited to a fixed voxel grid of $51.2\,\text{m} \times 51.2\,\text{m} \times 6.4\,\text{m}$. Given that our point cloud generation is done over a scan with a radius of $50\,\text{m}$, we divide the input scan into four quadrants over the $360°$ LiDAR field of view, generating the complete scene over each quadrant and finally gathering them together as the final prediction. All baselines and our method were trained only with SemanticKITTI, and later evaluated on SemanticKITTI, and on KITTI-360 and our data without fine-tuning.

4.1. Scene reconstruction

In this experiment we evaluate how close is the predicted scan completion from the expected complete scene. To do so, we quantify it with two metrics, the Chamfer distance (CD) and the Jensen-Shannon divergence (JSD). The Chamfer distance evaluates the completion at point level, measuring the level of detail of the generated scene by calculating how far are its points from the expected scene. The JSD compares the points distribution between the generated and the ground truth scene. For the JSD, we follow the evaluation done by Xiong *et al*. [42], where the scene is first voxelized with a grid resolution of $0.5\,\text{m}$ and then projected to a bird's eye view (BEV) evaluating over this projection.

Tab. 1 shows the results comparing our approach with previous state-of-the-art methods, where our method achieves the best performance in both metrics. First, we can notice that the state-of-the-art shape generation diffusion method, PVD, achieves the lowest performance, showing that current 3D diffusion methods cannot directly be applied to scene-scale data. The best performance of our method over the CD metric can be explained by the fact that our method operates directly on the points, which enables it to produce a more detailed scene compared to the baselines. The scene representation from SDF-based methods inherits artifacts from the surface approximation and voxelization, impacting the details in the reconstructed scene and therefore decreasing their performance with respect to the CD. The JSD evaluates the reconstructed scene points distribution over a voxelized grid comparing the overall scene distribution between the generated and the expected completion. Even though our method is not optimized over a voxel representation, we still achieve the best performance, showing that our scene completion is at the same time closer to the expected point distribution and can yield more details.

Tab. 2 compares the results of the scene completion methods on KITTI-360 and our collected data. Due to the poor performance of PVD over KITTI dataset, we do not evaluate it on those datasets. For KITTI-360 we notice the same behavior as in Tab. 1, where our method achieves the best performance in both metrics. When evaluating in our data, the performance of the SDF-based methods improve. This is expected since our data has denser point clouds, which is an advantage for such methods since they rely on the input points to approximate a surface to represent the scene. However, our method still achieves the best performance on the JSD metric and competitive performance on the CD metric. This evaluation shows that our method can still achieve scene completion over different datasets without fine-tuning since its generation is conditioned to the input scan. In Fig. 5 we can compare the scene completion generation between the methods. We can see that the diffusion baseline, PVD, fails on generating scene-scale data. SDF-based methods inherits artifacts from the voxelization, while our method, especially after the refinement, can generate a scene closer to the expected, following closely the structural information from the input scan.

4.2. Scene occupancy

In this experiment, we assess the scene completion by evaluating the occupancy of the predicted scene compared with the ground truth. To do so, we follow the evaluation proposed by Song *et al*. [38] where the intersection-over-union

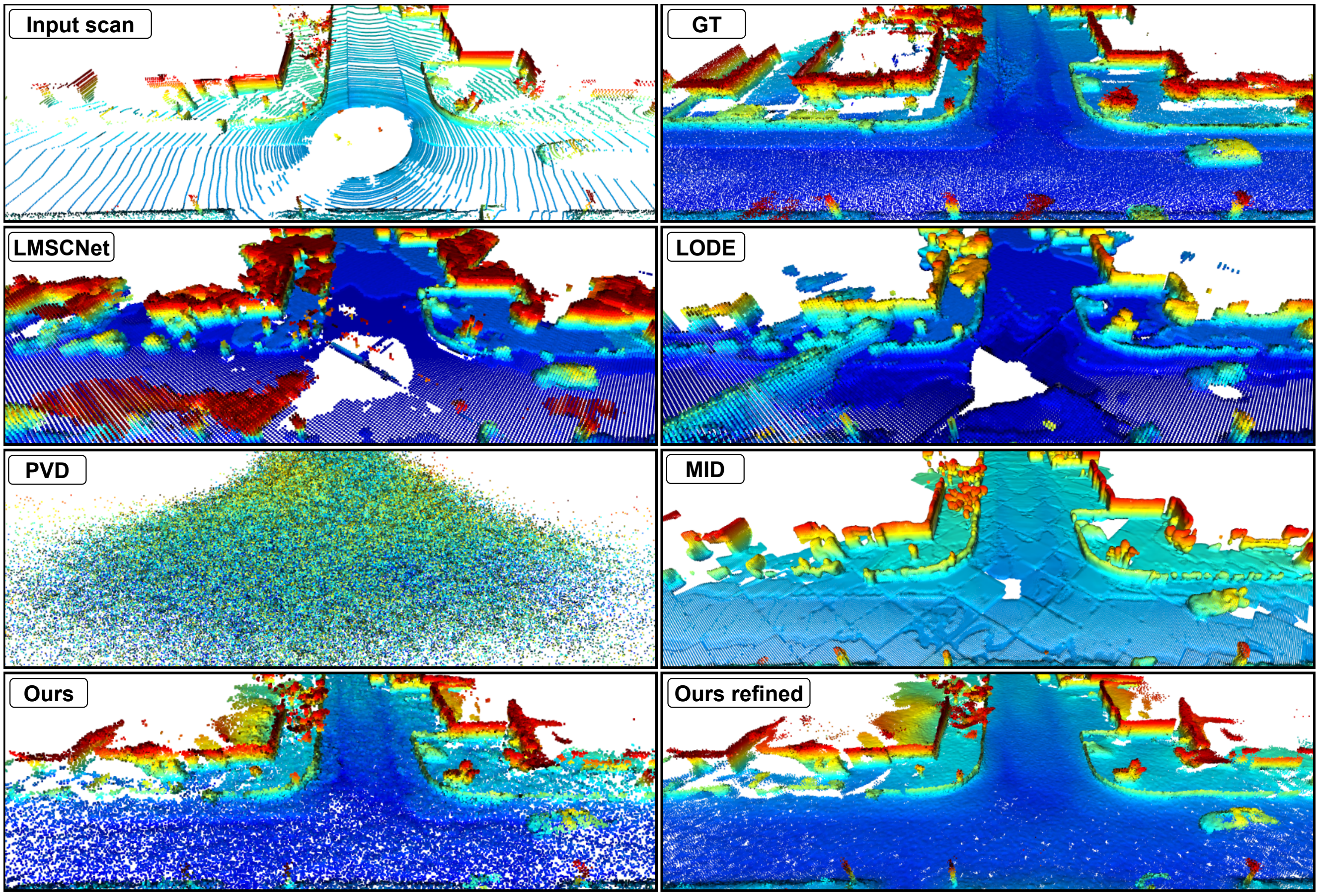

Figure 5. Qualitative results on one scan from KITTI-360. Colors depict point height normalized by the height range of each point cloud.

	IoU [%] Grid resolution (m)		
Method	0.5	0.2	0.1
LMSCNet [32]	32.23	23.05	3.48
LODE [15]	43.56	**47.88**	6.06
MID [40]	**45.02**	41.01	16.98
PVD [47]	21.20	7.96	1.44
Ours	42.49	33.12	11.02
Ours refined	40.71	38.92	**24.75**

Table 3. Completion metric where the IoU is computed against the ground truth and prediction grids with different resolutions.

	KITTI-360 (IoU) [%] Grid resolution (m)			Our data (IoU) [%] Grid resolution (m)		
Method	0.5	0.2	0.1	0.5	0.2	0.1
LMSCNet [32]	25.46	16.35	2.99	21.93	8.48	0.95
LODE [15]	42.08	**42.63**	5.85	42.99	42.24	5.14
MID [40]	**44.11**	36.38	15.84	**44.47**	**44.08**	16.38
Ours	42.22	32.25	10.80	37.16	29.17	6.53
Ours refined	40.82	36.08	**21.34**	38.51	40.20	**17.48**

Table 4. Completion metric where the IoU is computed against the ground truth and prediction grids with different resolutions.

(IoU) is computed between the predicted and ground truth voxelized scene, classifying each voxel as occupied or not. In this evaluation, we compute the IoU at three different voxel resolutions, *i.e.*, 0.5 m, 0.2 m, and 0.1 m. With a voxel size of 0.5 m, we evaluate the occupancy over the coarse scene, where the scene details are not considered. As we decrease the voxel size, more fine-grained details are considered in the evaluation.

Tab. 3 shows the IoU of our method compared to the baselines at the different voxel resolutions. First, the diffusion baseline PVD has the lowest performance overall. This again shows that current state-of-the-art 3D shape completion diffusion methods cannot be directly applied to scene-scale data. At a higher voxel size, our approach stays behind some SDF-based baselines. This is reasonable in this evaluation since SDF methods use a voxel representation to reconstruct the scene. Therefore, its reconstruction is equally distributed over the point cloud and its voxel representation is denser compared to our result voxelized. As we decrease the voxel size, the baselines performance drops. At the lowest resolution, our method outperforms the baselines. LM-

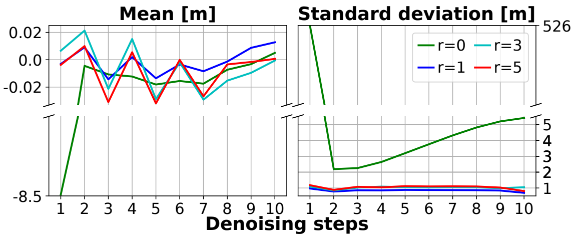

Figure 6. Mean and standard deviation of the predicted noise ϵ_θ over different regularization weights. In this experiment we use DPMSolver [17] to reduce the denoising steps from $1,000$ to 10.

SCNet [32] and LODE [15] are bound to a voxel resolution of $0.2\,\text{m}$, therefore with a voxel size of $0.1\,\text{m}$ their performance drops drastically. Make It Dense [40] was trained with a voxel size of $0.1\,\text{m}$, however, our method still outperforms it at this resolution. This shows the advantage of our approach. Since it is trained at point level, it can produce a more detailed scene, not limited to a fixed grid size.

Due to the poor performance of PVD on SemanticKITTI, we compared our method only with non-diffusion approaches for the other two datasets. In Tab. 4, the same behavior is seen on KITTI-360 and our data. At higher voxel resolution, the SDF baselines have a higher IoU, while with a lower voxel size, our method achieves the best performance. It is also noteworthy that despite of SDF-based having advantage in our data as discussed in Sec. 4.1, our method still achieves the best performance at lower resolution. This suggests that our approach can reconstruct the scene with more details, and it is able to generate data from a different dataset than the one it was trained on, since its generation is guided by the input LiDAR scan.

4.3. Noise regularization

In this section, we evaluate the impact of the proposed noise prediction regularization on the generated scene. We compare the predicted noise distribution with different regularization weights r in Fig. 6 from the 10 denoising steps in one scan as in Fig. 3. As can be seen, without the regularization, *i.e.*, $r = 0$, the predicted noise starts far from the expected distribution, with a mean of around -9.0 and a standard deviation of about 526. As we denoise the input, the distribution gets closer to the expected, however, still with a high standard deviation. When we add our proposed regularization, the model already starts predicting a more reasonable noise distribution from the beginning, stabilizing the denoising process. From this evaluation, we noticed that using $r = 5.0$ achieved a more stable distribution over the denoising steps. In our supplementary material, we provide also qualitative comparison between the generated point clouds with different regularization weights.

To evaluate how the regularization impacts the data generation, we compare the model performance over a short sequence from the SemanticKITTI validation set. We run the

r	0.0	1.0	3.0	5.0
CD [m]	0.529	0.470	0.441	0.445

Table 5. Mean chamfer distance over a short sequence from the validation set of SemanticKITTI.

scene completion pipeline every one hundred scans without using the refinement network, evaluating only the regularization influence over the noise predictor. In Tab. 5, we compute the chamfer distance to compare the impact of the regularization over the quality of the generated scene. As we increase the regularization, the generation quality improves. Despite $r = 3.0$ achieving a slightly better result in this evaluation, we stick with $r = 5.0$ due to the analysis of the noise distribution from Fig. 6, and from the qualitative comparisons provided in the supplementary material.

5. Conclusion

In this paper, we propose a novel point-level denoising diffusion probabilistic model to achieve scene completion using autonomous driving data. We exploit the generative capabilities of DDPMs to generate the missing parts from a single sparse LiDAR scan. We reformulate the diffusion process as a local problem. We define each point as the origin of the sampled Gaussian noise, learning an iterative denoising process to gradually predict offsets to reconstruct the scene from the input noisy LiDAR scan. This formulation enables the processing of scene-scale 3D data, retaining more details during the denoising process. In our experiments, we compare our method with recent state-of-the-art diffusion and non-diffusion methods. Our results show that our approach produces a more fine-grained completion compared to the baselines and can achieve scene completion on different datasets since its generation is conditioned to the input LiDAR scan. Besides, our proposed diffusion formulation distinguishes from previous state-of-the-art diffusion approaches by enabling the generation of scene-scale 3D data. Furthermore, we believe that our scene-scale diffusion formulation can support further research in the 3D diffusion generation research field.

Limitations. Even though achieving compelling results on scene completion, our method is still not able to generate unconditional data. This limits the data generation capability since it requires an input scan to guide the generation. In our supplementary material, we show examples of the unconditional generation of our approach. For future work, we plan on extending our method to generate unconditional data, creating novel 3D point cloud scenes.

Acknowledgments. This work has partially been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy, EXC-2070 – 390732324 – PhenoRob.

References

[1] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, Tero Karras, and Ming-Yu Liu. ediff-i: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint*, arXiv:2211.01324, 2022. 2

[2] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Juergen Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2019. 2, 5

[3] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Juergen Gall, and Cyrill Stachniss. Towards 3D LiDAR-based Semantic Scene Understanding of 3D Point Cloud Sequences: The SemanticKITTI Dataset. *Intl. Journal of Robotics Research (IJRR)*, 40(8–9):959–967, 2021.

[4] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A Multimodal Dataset for Autonomous Driving. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[5] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 5

[6] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2021. 2, 3

[7] Whye Kit Fong, Rohit Mohan, Juana Valeria Hurtado, Lubing Zhou, Holgr Caesar, Oscar Beijbom, and Abhinav Valada. Panoptic nuScenes A Large-Scale Benchmark for LiDAR Panoptic Segmentation and Tracking. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2022. 2

[8] Chen Fu, Chiyu Dong, Christoph Mertz, and John M. Dolan. Depth Completion Via Inductive Fusion of Planar LIDAR and Monocular Camera. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020. 1, 2

[9] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012. 2, 5

[10] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. 2, 3

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2020. 2, 3

[12] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2022. 2, 5

[13] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2015. 5

[14] Jumin Lee, Woobin Im, Sebin Lee, and Sung-Eui Yoon. Diffusion probabilistic models for scene-scale 3d categorical data. *arXiv preprint*, arXiv:2301.00527, 2023. 2

[15] Pengfei Li, Ruowen Zhao, Yongliang Shi, Hao Zhao, Jirui Yuan, Guyue Zhou, and Ya-Qin Zhang. LODE Locally Conditioned Eikonal Implicit Scene Completion from Sparse LiDAR. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2023. 1, 2, 6, 7, 8

[16] Yiyi Liao, Jun Xie, and Andreas Geiger. KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 2022. 2, 5

[17] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2022. 2, 4, 5, 8

[18] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint*, arXiv:2211.01095, 2023. 2, 5

[19] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 3

[20] Zhaoyang Lyu, Zhifeng Kong, Xudong XU, Liang Pan, and Dahua Lin. A conditional point diffusion-refinement paradigm for 3d point cloud completion. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2022. 2, 3, 4, 5

[21] Fangchang Ma and Sertac Karaman. Sparse-To-Dense: Depth Prediction from Sparse Depth Samples and a Single Image. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018. 1, 2

[22] Fangchang Ma, Guilherme Venturelli Cavalheiro, and Sertac Karaman. Self-Supervised Sparse-To-Dense Self-Supervised Depth Completion from LiDAR and Monocular Camera. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2019. 1, 2

[23] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 5

[24] Benedikt Mersch, Tiziano Guadagnino, Xieyuanli Chen, Tiziano, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. Building Volumetric Beliefs for Dynamic Environments Exploiting Map-Based Moving Object Segmentation. *IEEE Robotics and Automation Letters (RA-L)*, 8(8):5180–5187, 2023. 5

[25] Zak Murez, Tarrence van As, James Bartolozzi, Ayan Sinha, Vijay Badrinarayanan, and Andrew Rabinovich. Atlas: End-to-end 3d scene reconstruction from posed images. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2020. 1

[26] Kazuto Nakashima and Ryo Kurazume. Lidar data synthesis with denoising diffusion probabilistic models. *arXiv preprint*, arXiV:2309.09256, 2023. 2

[27] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *Proc. of Machine Learning Research (PMLR)*, 2021. 2, 3

[28] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2023. 2, 3

[29] Marija Popović, Florian Thomas, Sotiris Papatheodorou, Nils Funk, Teresa Vidal-Calleja, and Stefan Leutenegger. Volumetric Occupancy Mapping With Probabilistic Depth Completion for Robotic Navigation. *IEEE Robotics and Automation Letters (RA-L)*, 6(3):5072–5079, 2021. 1, 2

[30] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *Proc. of the International Conference on Machine Learning*, 2021. 2, 3

[31] Christoph Rist, David Emmerichs, Markus Enzweiler, and Dariu Gavrila. Semantic scene completion using local deep implicit functions on lidar data. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 44(10):7205–7218, 2021. 2

[32] Luis Roldão, Raoul de Charette, and Anne Verroust-Blondet. LMSCNet: Lightweight Multiscale 3D Semantic Completion. In *Proc. of the Intl. Conf. on 3D Vision (3DV)*, 2020. 2, 6, 7, 8

[33] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis With Latent Diffusion Models. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 3

[34] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2022. 2, 5

[35] Aditya Sanghi, Hang Chu, Joseph G. Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshan. CLIP-Forge: Towards Zero-Shot Text-To-Shape Generation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 3

[36] Aditya Sanghi, Rao Fu, Vivian Liu, Karl D.D. Willis, Hooman Shayani, Amir H. Khasahmadi, Srinath Sridhar, and Daniel Ritchie. CLIP-Sculptor: Zero-Shot Generation of High-Fidelity and Diverse Shapes From Natural Language. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 3

[37] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2021. 2, 5

[38] Shuran Song, Fisher Yu, Andy Zeng, Angel X. Chang, Manolis Savva, and Thomas Funkhouser. Semantic Scene Completion from a Single Depth Image. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 6

[39] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Sheng Zhao, Shuyang Cheng, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[40] Ignacio Vizzo, Benedikt Mersch, Rodrigo Marcuzzi, Louis Wiesmann, Jens Behley, and Cyrill Stachniss. Make it dense: Self-supervised geometric scan completion of sparse 3d lidar scans in large outdoor environments. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):8534–8541, 2022. 1, 2, 6, 7, 8

[41] Ignacio Vizzo, Tiziano Guadagnino, Benedikt Mersch, Louis Wiesmann, Jens Behley, and Cyrill Stachniss. KISS-ICP: In Defense of Point-to-Point ICP – Simple, Accurate, and Robust Registration If Done the Right Way. *IEEE Robotics and Automation Letters (RA-L)*, 8(2):1029–1036, 2023. 5

[42] Yuwen Xiong, Wei-Chiu Ma, Jingkang Wang, and Raquel Urtasun. Learning Compact Representations for LiDAR Completion and Generation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2, 6

[43] Jiale Xu, Xintao Wang, Weihao Cheng, Yan-Pei Cao, Ying Shan, Xiaohu Qie, and Shenghua Gao. Dream3D: Zero-Shot Text-to-3D Synthesis Using 3D Shape Prior and Text-to-Image Diffusion Models. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 3

[44] Yan Xu, Xinge Zhu, Jianping Shi, Guofeng Zhang, Hujun Bao, and Hongsheng Li. Depth Completion From Sparse LiDAR Data With Depth-Normal Constraints. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2019. 1, 2

[45] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2022. 2, 3

[46] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2023. 2

[47] Linqi Zhou, Yilun Du, and Jiajun Wu. 3D Shape Generation and Completion Through Point-Voxel Diffusion. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021. 2, 3, 4, 5, 6, 7

[48] Yufan Zhou, Ruiyi Zhang, Changyou Chen, Chunyuan Li, Chris Tensmeyer, Tong Yu, Jiuxiang Gu, Jinhui Xu, and Tong Sun. Towards Language-Free Training for Text-to-Image Generation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 3

[49] Yufan Zhou, Bingchen Liu, Yizhe Zhu, Xiao Yang, Changyou Chen, and Jinhui Xu. Shifted Diffusion for Text-to-Image Generation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 3

[50] Vlas Zyrianov, Xiyue Zhu, and Shenlong Wang. Learning to generate realistic lidar point clouds. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2022. 2

Certificate of Reproducibility

The authors of this publication declare that:

1. The software related to this publication is distributed in the hope that it will be useful, support open research, and simplify the reproducability of the results but it comes without any warrenty and without even the implied warranty of merchantability or fitness for a particular purpose.
2. *Lucas Nunes* primarily developed the implementation related to this paper. This was done on Ubuntu 22.04.
3. *Rodrigo Marcuzzi* verified that the code can be executed on a machine that follows the software specification given in the Git repository available at:

   https://github.com/PRBonn/LiDiff

4. *Rodrigo Marcuzzi* verified that the experimental results presented in this publication can be reproduced using the implementation used at submission, which is labeled with a tag in the Git repository and can be retrieved using the command:

   ```
   git checkout CVPR24
   ```