Diffusion-Based Signed Distance Fields for 3D Shape Generation

Jaehyeok Shim, Changwoo Kang and Kyungdon Joo*
Artificial Intelligence Graduate School, UNIST
{jh.shim,kangchangwoo,kyungdon}@unist.ac.kr
https://kitsunetic.github.io/sdf-diffusion

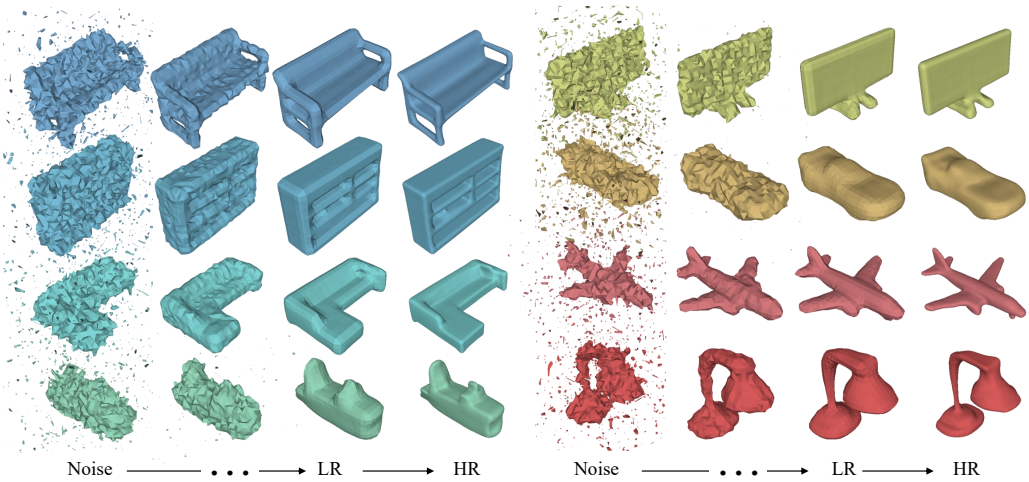

Figure 1. **Visualization of 3D shape generation for various categories using `SDF-Diffusion`**. `SDF-diffusion` gradually removes the Gaussian noise and generates high-resolution 3D shapes in the form of an SDF voxel by a two-stage framework. Unlike previous methods, which use point clouds, we can generate 3D meshes without concern of complex post-processing. For visualization purposes, we show less-noise data instead of the initial Gaussian noise.

Abstract

We propose a 3D shape generation framework (`SDF-Diffusion` in short) that uses denoising diffusion models with continuous 3D representation via signed distance fields (SDF). Unlike most existing methods that depend on discontinuous forms, such as point clouds, `SDF-Diffusion` generates high-resolution 3D shapes while alleviating memory issues by separating the generative process into two-stage: generation and super-resolution. In the first stage, a diffusion-based generative model generates a low-resolution SDF of 3D shapes. Using the estimated low-resolution SDF as a condition, the second stage diffusion model performs super-resolution to generate high-resolution SDF. Our framework can generate a high-fidelity 3D shape despite the extreme spatial complexity. On the ShapeNet dataset, our model shows competitive performance to the state-of-the-art methods and shows applicability on the shape completion task without modification.

*Corresponding author.

1. Introduction

The need for generative modeling of 3D shapes has rapidly increased due to high demand in various fields, such as computer vision, graphics, robotics, and content generation. To synthesize a high-quality 3D shape, numerous generative approaches have been actively studied, including generative adversarial networks (GAN) [1, 7, 8, 18, 20, 40, 51, 62, 71, 85], variational auto-encoders (VAE) [13, 27, 35, 45, 67], normalizing flows [26, 30, 60, 77], auto-regressive models [19, 31, 45, 46, 76], and others [61, 74, 75, 80, 84].

Recently, denoising diffusion models (DDM) have emerged as a promising generative framework in image generation [10, 16, 22, 39, 47, 63] and speech synthesis [5, 32, 54]. DDM achieves a generative process by gradually corrupting data through a diffusion process and denoising through a learned neural network. This network can generate new realistic data by repeating the denoising process from the given pure noise. Based on the success of DDM-based generative models in various domains, several attempts are being made to apply them to the task of 3D shape generation [41, 83, 87]. They have applied DDM to generate new point cloud and have outperformed previous methods.

However, even though DDM-based 3D generative models have demonstrated their impressive performance, they still have limitations to being adopted for real-world problems. Most previous DDM-based 3D generative models use point clouds to represent 3D shapes, which limits their ability to express continuous surfaces of 3D shapes, unlike mesh representation commonly used in real-world applications. To resolve this issue, intricate post-processing [2, 24, 25, 52] is required to reconstruct continuous meshes from point clouds. They require various parameter tuning or additional information (*e.g.*, surface normal at each point [52]), and a densely sampled point cloud is required to reconstruct high-quality mesh. However, existing DDM-based methods for 3D shape generation are based on sparse point clouds.

In this work, we introduce a new DDM-based generation framework that generates high-quality 3D shapes through signed distance fields (SDF) (see Fig. 1). We can view SDF as a function that takes an arbitrary location as input and returns a signed distance value from the input location to the nearest surface of the mesh, and the sign of the value means whether inside or outside of the shape. We sample SDF values uniformly from a 3D shape to form a voxel-shaped SDF. This form has several advantages over point clouds. It can directly reconstruct mesh through the marching cube algorithm [38] and can utilize convolutional neural network (CNN) because of its dense and fixed structure.

Based on this voxel-shaped SDF representation, we propose a novel DDM-based generative model in two-stage (see Fig. 2). In the first stage, a diffusion-based SDF generation produces low-resolution SDF for coarse 3D shapes. In the second stage, we propose a diffusion-based SDF super-resolution, given the low-resolution SDF of the first stage as a condition. In particular, since the increasing resolution of the voxel comes as a significant burden in terms of computational resources (*i.e.*, due to the curse of the dimension), we approach this problem by super-resolution in patch-based learning. Under this scheme, we can iteratively perform the second stage multiple times with the same structure and generate further higher resolution of SDF, which is a more detailed 3D shape. With various experiments, we achieve state-of-the-art in single-/multi-category 3D shape generative quality, and our method can be directly converted into 3D meshes, unlike other point cloud-based methods that require intricate post-processing.

In summary, our contributions are as follows:

- We propose a novel DDM-based generation method that utilizes a voxel-shaped SDF representation to generate high-quality and continuous 3D shapes.

- We represent a memory-efficient two-stage framework composed of low-resolution SDF generation and SDF super-resolution conditioned on the low-resolution SDF. In particular, we can iteratively perform super-resolution to generate higher-resolution SDF.

- `SDF-diffusion` can be seamlessly extended into multi-category 3D shape generation and completion, which shows the flexibility of the proposed method.

2. Related Work

3D Shape representation. Numerous in-depth studies have been conducted on the representation of 3D shape [8, 43, 44, 48–50, 64, 73, 78, 79, 81]. We can categorize them into two groups: explicit representations, such as voxels [14, 15, 71], point clouds [1, 55, 56], and implicit functions, such as signed distance fields (SDF) [50, 64].

Voxels are widely adapted in various neural networks due to their intuitive structure and synergy with 3D CNN [14, 15, 71, 73]. However, it may cause a memory problem according to the increase in resolution. Point clouds, which are compact and readily available, are dominant representations with their well-suitability to learn contextual information in neural networks [1, 55, 56, 79, 81]. However, it requires post-processing to convert the synthesized point clouds into meshes for real-world applications [8, 38, 43, 44, 48–50, 78].

Implicit functions view the 3D shape as a set of levels for a continuous function, which makes it possible to express a continuous surface [8, 44, 48–50, 64, 78]. As a result, despite their slow interference as they require numerous network operations, these methods are growing in popularity. In particular, SDF encoded with distance to the nearest surface has some advantages to express sharper reconstructions [50, 64]. Thus, we exploit SDF voxel representation to obtain sharper meshes without unnecessary procedures.

3D shape generation. To generate 3D geometry, a variety of frameworks are applied [1, 7, 8, 13, 19, 20, 26, 27, 30, 31, 34, 46, 51, 60, 62, 67, 85]. With such a perspective, several frameworks are applied to model the distribution representing a 3D shape, such as GAN-based models [33], auto-regressive models [66], flow-based models [77], and implicit functions [43, 50].

For GAN-based 3D generative method, SDF-StyleGAN [86] extends StyleGAN2 [23] for 3D shape generation by proposing global and local discriminator that enables GAN discriminator on SDF. Generative methods, such as AutoSDF [45], and ShapeFormer [76], propose to utilize a voxel form of implicit representation (*e.g.*, SDF and occupancy) for mesh generation through a two-stage. A two-stage generative strategy reduces required computational resources by compressing data into a tractable form (*e.g.*, latent feature) rather than directly dealing with an original data structure to make a generative model concentrate semantically compressed representation and save computational resources. Motivated by VQ-GAN [12],

AutoSDF and ShapeFormer apply two-stage generative method and show impressive performance on conditional 3D shape generation. They utilize an auto-encoder to convert voxel into compressed latent space and generate a new shape using the autoregressive model on the latent space. However, they miss fully utilizing the advantage of the two-stage generative method because they use a regression-based auto-encoder unlike VQ-GAN, which uses GAN and perceptual loss [21]. Unlike the 2D image generation, there are insufficient research works in perceptually plausible auto-encoding methods about 3D shapes, especially voxel structures; it has difficulties in applying a two-stage strategy to 3D shape generation.

On the other hand, recent research works propose a new two-stage generative method [17, 42, 59]. They use a super-resolution model to up-sample low-resolution data, and the generation model learns low-resolution data distribution. Instead of training an auto-encoder, these methods utilize naturally constructed low-resolution forms. Inspired by this, our method exploits a two-stage framework using voxel-shaped SDF. Instead of learning the auto-encoder, we propose a method of generating a coarse 3D shape that is naturally compressed form in the representation of a low-resolution voxel. We then generate high-quality 3D shapes by refining the details of the coarse shape by upsampling the low-resolution voxel.

Diffusion-Based 3D shape generation. With the success of DDM-based generative models in image generation [10, 16, 47], diffusion-based 3D shape generations have advanced significantly. In particular, diffusion probabilistic models (DPM) [41] suggest a DDM-based approach with an auto-encoder architecture built on PointNet [55], which compresses point clouds into a latent vector and then generates new shape latent using normalizing flows [6, 11]. Point voxel diffusion (PVD) [87] applies DDM directly on the point clouds by adopting a point voxel CNN-based network architecture [36], which utilizes both the 3D CNN and PointNet. In addition, point diffusion-refinement (PDR) [42] proposes a two-stage generative framework for conditional point cloud generation task. It generates sparse point clouds by PointNet-based DDM in the first stage and up-samples the sparse point clouds to denser ones in a learning-based manner. Motivated by latent diffusion models (LDM) [58], latent point diffusion models (LION) [83] propose a two-stage approach with compressed latent point clouds through VAE [29]. LION converts result point clouds into mesh by shapes as points (SAP) [52]. Despite their great performance, it still requires a pre-trained SAP model or fine-tuning stage. In other words, the usability of the above point clouds-based approaches is limited to be adopted in real-world applications. To address this issue, we propose a diffusion-based generative model upon voxel-shaped SDF representation, which can directly reconstruct 3D mesh using zero iso-surface finding methods [38].

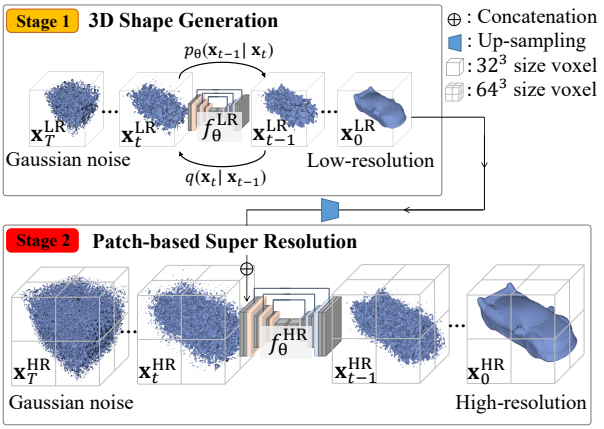

Figure 2. **Overview of `SDF-Diffusion`**. The proposed method consists of two stages. In the first stage, we take as input voxel-shaped noise $\mathbf{x}_T^{\mathrm{LR}} = \epsilon^{\mathrm{LR}}$ and generate low-resolution SDF voxel $\mathbf{x}_0^{\mathrm{LR}}$ of new coarse 3D shape through DDM. In the second stage, conditioned on $\mathbf{x}_0^{\mathrm{LR}}$, we perform super-resolution through another DDM, which generates high-resolution SDF voxel $\mathbf{x}_0^{\mathrm{HR}}$ with fine and detailed 3D shape.

3. Methodology

We propose a generative method for high-resolution 3D shapes in the form of SDF voxels based on the denoising diffusion models [16, 63] (`SDF-Diffusion` in short). To generate high-resolution samples efficiently, the proposed method involves two stages: diffusion-based SDF generation and diffusion-based patch-wise SDF super-resolution (see Fig. 2 for the overview). In the first stage, the diffusion-based SDF generation model generates a low-resolution coarse 3D shape. Given this low-resolution sample, the diffusion-based SDF super-resolution model progressively produces high-resolution samples in the second stage. In other words, we can consider the first stage as a pure diffusion-based generative model, and the second stage as a conditional diffusion-based super-resolution model. It is worth noting that, ideally, the second stage can be repeated multiple times with the same architecture, which enables us to generate more high-resolution 3D shapes.

In the following, we briefly explain the background on DDM in Sec. 3.1. Then, in Sec. 3.2, we describe our method for generating coarse 3D shapes using a diffusion-based technique. In Sec. 3.3, we explain how we generate high-resolution 3D shapes by super-resolving low-resolution correspondence. Training and inference processes for each diffusion model are listed in Algorithm 1 and Algorithm 2.

3.1. Background on Denoising Diffusion Models

DDM is a class of generative models that generates new samples from a given data distribution by learning the in-

verse of the noising process [16, 63]. Concretely, starting from samples from the Gaussian distribution $\mathbf{x}_T$, the denoising process gradually produces samples with less noise $\mathbf{x}_{T-1}, \mathbf{x}_{T-2}, \ldots$, and aims to reach final samples $\mathbf{x}_0$, which is the noise-free final output.

Forward process. The *forward process* (or *diffusion process*) gradually adds the Gaussian noise to the clean data, which is the approximate joint posterior $q(\mathbf{x}_{1:T}|\mathbf{x}_0)$. This diffusion process is represented as a Markov chain:

$$q(\mathbf{x}_{0:T}) = q(\mathbf{x}_0)\prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}), \tag{1}$$

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}), \tag{2}$$

where β_t denotes the noise scheduling hyperparameter.

According to [16], the forward process admits sampling $\mathbf{x}_t$ at an arbitrary time step t by a closed form:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}), \tag{3}$$

where $\alpha_t = 1-\beta_t$ and $\bar{\alpha}_t = \Pi_{s=1}^{t}\alpha_s$. Thus, $\mathbf{x}_t$ can be directly sampled by:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{4}$$

Reverse process. The *reverse process* (or *generative process*) aims to generate samples by gradually inverting infinitesimal noise, which is the joint distribution $p_\theta(\mathbf{x}_{0:T})$. This reverse process is defined as a Markov chain with the learned Gaussian transitions:

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t), \tag{5}$$

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_t^2\mathbf{I}), \tag{6}$$

where $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$ is the standard Gaussian distribution, $\mu_\theta(\mathbf{x}_t, t)$ indicates a denoising function parameterized by θ, σ_t^2 is a time-step dependant variance, and we fix $\sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$, following [16]. We can sample from $p(\mathbf{x}_T)$ and then draw $\mathbf{x}_{t-1} \sim p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ by Eq. (6) as:

$$\mathbf{x}_{t-1} = \mu_\theta(\mathbf{x}_t, t) + \sigma_t\epsilon, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{7}$$

By repeating this process, we can eventually generate $\mathbf{x}_0$.

Training objective. Following [16], we can simplify $\mu_\theta(\mathbf{x}_t, t)$:

$$\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\right), \tag{8}$$

where $\epsilon_\theta(\mathbf{x}_t, t)$ is a neural network parameterized by θ that predicts noise from noisy input $\mathbf{x}_t$. Thus, the objective function compares the difference between the predicted noise $\epsilon_\theta(\mathbf{x}_t, t)$ and the applied noise ϵ [16]:

$$\mathcal{L} = \|\epsilon_\theta(\mathbf{x}_t, t) - \epsilon\|_p^p, \tag{9}$$

where $p \in \{1, 2\}$. On the other hand, according to [16], it is possible to modify $\mu_\theta(\mathbf{x}_t, t)$ with a neural network predicting $\mathbf{x}_0$ instead of noise ϵ:

$$\mu_\theta(\mathbf{x}_t, t) = \gamma_t f_\theta(\mathbf{x}_t, t) + \delta_t\mathbf{x}_t, \tag{10}$$

where $\gamma_t = \frac{\beta_t\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_t}$ and $\delta_t = \frac{(1-\bar{\alpha}_{t-1})\sqrt{\alpha_t}}{1-\bar{\alpha}_t}$, and $f_\theta(\mathbf{x}_t, t)$ is a neural network parameterized by θ that predicts noise-free data $\mathbf{x}_0$. In that case, the objective function becomes:

$$\mathcal{L} = \|f_\theta(\mathbf{x}_t, t) - \mathbf{x}_0\|_p^p. \tag{11}$$

Thus, we use Eq. (11) as our objective function in this work.

3.2. Diffusion SDF Generation

Here, we describe a diffusion-based generative model for low-resolution 3D shape represented as SDF voxel $\mathbf{x}_0^{\text{LR}} \sim q(\mathbf{x}_0^{\text{LR}})$ sampled from the real data distribution $q(\mathbf{x}_0^{\text{LR}})$.

Diffusion-based SDF voxel generation. Specifically, given a clean data sample $\mathbf{x}_0^{\text{LR}}$, we corrupt $\mathbf{x}_0^{\text{LR}}$ with the same shaped voxel noise $\epsilon^{\text{LR}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ sampled from the standard Gaussian distribution following the forward process in Eq. (4):

$$\mathbf{x}_t^{\text{LR}} = \sqrt{\bar{\alpha}_t}\mathbf{x}_0^{\text{LR}} + \sqrt{1-\bar{\alpha}_t}\epsilon^{\text{LR}}, \epsilon^{\text{LR}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{12}$$

Then, a network f_θ^{LR} is trained to predict noise-free data with MSE objective function like Eq. (11):

$$\mathcal{L}^{\text{LR}} = \|f_\theta^{\text{LR}}(\mathbf{x}_t^{\text{LR}}, t, c) - \mathbf{x}_0^{\text{LR}}\|_2^2, \tag{13}$$

where c is a category label of given data.

During inference, a new SDF voxel is generated by progressively predicting lesser noisy samples $\mathbf{x}_{t-1}^{\text{LR}}$ with the trained neural network f_θ^{LR} as $t = T, \ldots, 1$:

$$\mathbf{x}_{t-1}^{\text{LR}} = \gamma_t f_\theta^{\text{LR}}(\mathbf{x}_t^{\text{LR}}, t, c) + \delta_t\mathbf{x}_t^{\text{LR}} + \sigma_t\epsilon^{\text{LR}}. \tag{14}$$

In Eq. (13) and Eq. (14), the category condition c is used only for multi-category generations.

Network architecture. Our model is built upon a U-shaped network in SR3 [59], which is a DDM-based super-resolution method designed for 2D image domain. Thus, we modify and improve this structure for 3D domain. We convert 2D-based U-shaped network of SR3 into 3D-based model by replacing 2D-based convolutional layers with 3D ones. The network has three layers, and each layer is composed of several layers of residual blocks and self-attention [68] and the Swish activation function [57]. The voxel feature resolution is halved for each layer (*e.g.*, $32 \rightarrow 16 \rightarrow 8$), and the channel is doubled (*e.g.*, $64 \rightarrow 128 \rightarrow 256$). A more detailed design of the network architecture is in the supplementary material.

The timestep t and category condition c are embedded into a vector $\mathbf{z}$ through MLPs following [4, 59]:

$$\mathbf{z} = \text{MLP}(\texttt{PosEnc}(\sqrt{\bar{\alpha}_t})) + \text{MLP}(\texttt{Embedding}(c)), \tag{15}$$

Algorithm 1 Training

Stage 1: Diffusion SDF Generation

repeat

$(\mathbf{x}_0^{\text{LR}}, c) \sim q(\mathbf{x}_0^{\text{LR}}, c)$ ▷ Sample LR voxel and category label

$t \sim \mathcal{U}(\{1, \dots T^{\text{LR}}\})$

$\epsilon^{\text{LR}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

$\mathbf{x}_t^{\text{LR}} = \sqrt{\bar{\alpha}_t}\mathbf{x}_0^{\text{LR}} + \sqrt{1-\bar{\alpha}_t}\epsilon^{\text{LR}}$ ▷ Noisy voxel at timestep t

$\theta \leftarrow \eta \nabla_\theta \| f_\theta^{\text{LR}}(\mathbf{x}_t^{\text{LR}}, t, c) - \mathbf{x}_0^{\text{LR}} \|_2^2$

until f_θ^{LR} is converged

Stage 2: Patch-Based Diffusion SDF Super-Resolution

repeat

$(\hat{\mathbf{x}}_0^{\text{HR}}, c) \sim q(\hat{\mathbf{x}}_0^{\text{HR}}, c)$ ▷ Sample HR voxel patch and category label

$\hat{\mathbf{x}}_0^{\text{LR}} \sim q(\hat{\mathbf{x}}_0^{\text{LR}})$ ▷ Sample LR conditional voxel patch

$t \sim \mathcal{U}(\{1, \dots T^{\text{HR}}\})$

$\hat{\epsilon}^{\text{HR}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

$\hat{\mathbf{x}}_t^{\text{HR}} = \sqrt{\bar{\alpha}_t}\hat{\mathbf{x}}_0^{\text{HR}} + \sqrt{1-\bar{\alpha}_t}\hat{\epsilon}^{\text{HR}}$ ▷ HR noisy voxel at timestep t

$\theta \leftarrow \eta \nabla_\theta \| f_\theta^{\text{HR}}(\hat{\mathbf{x}}_t^{\text{HR}}, \hat{\mathbf{x}}_0^{\text{LR}}, t, c) - \hat{\mathbf{x}}_0^{\text{HR}} \|_1^1$

until f_θ^{HR} is converged

where `PosEnc` is the sinusoidal positional encoding used in Transformer [68] and `Embedding` is the embedding layer, and the second term is ignored for a single-category generation. The embedding vector $\mathbf{z}$ is inserted into every residual block using adaptive group normalization layers (AdaGN) following [10]. AdaGN is an extension of group normalization [72] through channel-wise scaling and shifting *w.r.t.* normalized feature maps $\mathbf{h} \in \mathbb{R}^{k \times h \times w \times d}$, where k is the number of feature map channels:

$$\text{AdaGN}(\mathbf{h}, \mathbf{z}) = \mathbf{z}_s(\text{GroupNorm}(\mathbf{h}) + \mathbf{z}_t), \tag{16}$$

where $(\mathbf{z}_s, \mathbf{z}_t) \in \mathbb{R}^{2 \times k} = \text{MLP}(\mathbf{z})$ is result of an MLP layer. The detailed network architecture is available in the supplementary material.

3.3. Patch-Based Diffusion SDF Super-Resolution

We aim to train a conditional diffusion-based super-resolution model that generates realistic high-resolution SDF voxel $\mathbf{x}_0^{\text{HR}}$ with low-resolution condition $\mathbf{x}_0^{\text{LR}}$.

Patch-based learning scheme. The increase in demanding memory due to the growth of spatial complexity hinders the learning of high-resolution generative models. Instead of training on the full high-resolution voxel, we train our model based on small cube-shaped patches of given input high-resolution voxel $\mathbf{x}_0^{\text{HR}} \sim q(\mathbf{x}_0^{\text{HR}})$ sampled from real data distribution to reduce memory consumption. We denote $\hat{\mathbf{x}}_t^{\text{HR}}$ as a cube-shaped patch extracted from a random location of voxel $\mathbf{x}_t^{\text{HR}}$, and $\hat{\mathbf{x}}_t^{\text{LR}}$ is a patch extracted from the corresponding location of $\mathbf{x}_t^{\text{LR}}$. In addition, ϵ^{HR} and $\hat{\epsilon}^{\text{HR}}$ are voxel-shaped noises sampled from the standard Gaussian distribution that have the same shape with $\mathbf{x}_t^{\text{HR}}$ and $\hat{\mathbf{x}}_t^{\text{HR}}$ respectively. Note that the patch-based operation is optional during the test phase if GPU has enough memory.

Conditional diffusion-based super-resolution. Similarly to Eq. (12), the diffusion process for SDF super-resolution can be represented as:

$$\hat{\mathbf{x}}_t^{\text{HR}} = \sqrt{\bar{\alpha}_t}\hat{\mathbf{x}}_0^{\text{HR}} + \sqrt{1-\bar{\alpha}_t}\hat{\epsilon}^{\text{HR}}, \hat{\epsilon}^{\text{HR}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{17}$$

In the case of the objective function, we use $\mathcal{L}_1$ for super-resolution, followed by SR3 [59], where they empirically find that $\mathcal{L}_1$ loss is better for super-resolution task compared to MSE loss:

$$\mathcal{L}^{\text{HR}} = \| f_\theta^{\text{HR}}(\hat{\mathbf{x}}_t^{\text{HR}}, \hat{\mathbf{x}}_0^{\text{LR}}, t, c) - \hat{\mathbf{x}}_0^{\text{HR}} \|_1^1. \tag{18}$$

The low-resolution condition is nearestly up-sampled to be the same resolution as the high-resolution side and channel-wise concatenation is applied. Sampling is done by repeating a similar process to Eq. (14):

$$\mathbf{x}_{t-1}^{\text{HR}} = \gamma_t f_\theta^{\text{HR}}(\mathbf{x}_t^{\text{HR}}, \mathbf{x}_0^{\text{LR}}, t, c) + \delta_t \mathbf{x}_t^{\text{HR}} + \sigma_t \epsilon^{\text{HR}}, \tag{19}$$

In Eqs. (18) and (19), the category condition c is ignored for a single-category generation.

In addition, our model can generate further higher resolution shapes by iteratively applying the second stage model trained on multiple resolutions (*e.g.*, $32^3 \rightarrow 64^3$ and $64^3 \rightarrow 128^3$). We provide this higher-resolution generation in the experiments.

Algorithm 2 Sampling

Stage 1: Diffusion-Based 3D Shape Generation

Input: c: category label (integer)

Output: $\mathbf{x}_0^{\text{LR}}$: generated coarse shape

$t = T$

$\mathbf{x}_t^{\text{LR}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

while $t \neq 0$ **do**

$\epsilon^{\text{LR}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

$\mathbf{x}_{t-1}^{\text{LR}} \leftarrow \gamma_t f_\theta^{\text{LR}}(\mathbf{x}_t^{\text{LR}}, t, c) + \delta_t \mathbf{x}_t^{\text{LR}} + \sigma_t \epsilon^{\text{LR}}$

$t = t - 1$

end while

Stage 2: Patch-Based Diffusion SDF Super-Resolution

Input: c: category label (integer), $\mathbf{x}_0^{\text{LR}}$: low-resolution condition

Output: $\mathbf{x}_0^{\text{HR}}$: generated high-resolution shape

$t = T$

$\mathbf{x}_t^{\text{HR}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ ▷ Patch is optional during inference phase

while $t \neq 0$ **do**

$\epsilon^{\text{HR}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

$\mathbf{x}_{t-1}^{\text{HR}} \leftarrow \gamma_t f_\theta^{\text{HR}}(\mathbf{x}_t^{\text{HR}}, \mathbf{x}_0^{\text{LR}}, t, c) + \delta_t \mathbf{x}_t^{\text{HR}} + \sigma_t \epsilon^{\text{HR}}$

$t = t - 1$

end while

4. Experiments

In this section, we evaluate and compare the proposed method with state-of-the-art methods in 3D shape generations. Specifically, we present qualitative and quantitative comparisons of our method for single-/multi-category 3D shape generation tasks in Secs. 4.1 and 4.2, respectively. Additionally, we demonstrate the applicability of the proposed `SDF-Diffusion` for shape completion in Sec. 4.3. Before diving into the details, we first provide several pieces of information for experiments, such as implementation details, dataset, evaluation metrics, and comparison methods.

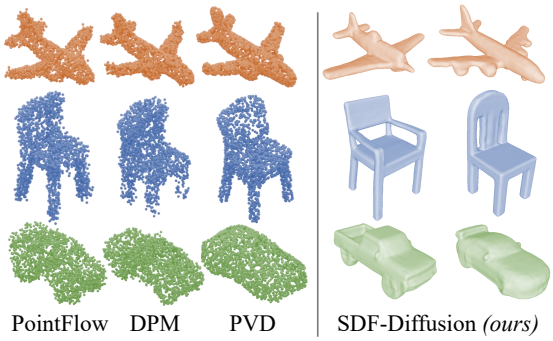

Figure 3. **Generated 3D shapes with a single-category generative model**. Baseline methods generate point clouds with $2,048$ points, and `SDF-Diffusion` produces 3D shapes in the form of T-SDF voxels, which can be converted to meshes directly through zero iso-surface finding algorithms.

Implementation details. For both generation and super-resolution models, we share the same network structure regardless of single-category and multi-category generations except for the category embedding layer (*i.e.*, the same network architecture is shared with all of our experiments). For point cloud input, we use the point cloud encoder from the convolutional occupancy network [53] as our point cloud encoder and combine the resulting features using AdaGN.

We use the Adam optimizer [28] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$, to train all of our models. We train our method with learning rate 0.0001 with learning rate warmup at the first epoch and reduce the learning rate when the validation loss is not decreased for 5 epochs. We fix the patch size to 32^3 for training the super-resolution model regardless of the target resolution. On the other hand, we sample the full resolution of the shape at once without using patches during the inference phase. Following [16], we set diffusion step $T = 1,000$, and linear noise scheduler with $\beta_0 = 0.0001$ and $\beta_T = 0.02$. We reduce sampling steps to 50 using the denoising diffusion implicit model (DDIM) [65] to accelerate sampling speed. We use the exponential moving average (EMA) following [23] for the first stage model. Flip data augmentation is applied to the super-resolution model.

Dataset. For evaluation, we use the ShapeNet dataset [3] (especially *version 1*), which consists of 13-category of 3D shapes in mesh representations. In comparison to prior work, our method requires sampled SDF in the training and evaluation phase. For this reason, we convert meshes in the ShapeNet dataset into voxel-shaped SDF similar to the dual octree graph networks [69]. Concretely, we normalize each of the watertight mesh so that it has zero center and unit scale with 0.1 padding. We then extract SDF voxels at multiple resolutions of 32 for low-resolution and 64 and 128 for high-resolution using Mesh2SDF [70]. While sampling watertight mesh, we manipulate the level set value, which is the parameter of Mesh2SDF, to approximately double cell size (*i.e.*, 0.03125 for voxel resolution of 64). We then clip each of the extracted SDF voxel with minimum and maximum thresholds as the same value with double voxel size, and we utilize SDF in the form of truncated signed distance fields (T-SDF) to decrease redundant information. After that, we sample 15K points from the watertight meshes at the resolution of 64, which are used as ground truth for training point cloud-based methods and evaluation purposes. Following [69], the dataset is split into the train, validation, and test datasets. We mainly measure the quality of our method with point clouds sampled from a voxel resolution of 64 in the validation dataset like previous methods.

Evaluation metrics. Following the previous methods [77, 87], we use three metrics (MMD, COV, and 1-NNA [37]) based on both the Chamfer Distance (CD) and the Earth Mover's Distance (EMD) as the main evaluation metrics. MMD, COV, and 1-NNA analyze both quality and variety by quantifying the distributional similarity between the produced forms and the validation set. A detailed explanation of the evaluation method is in the supplementary material.

Comparison methods. To evaluate the generative ability of our model, we compare it with various methods, such as DDM-based techniques: point voxel diffusion (PVD) [87] and diffusion point cloud (DPM) [41], as well as flow-based technique, PointFlow [77]. As most of the previous methods have been trained using the ShapeNet dataset pre-processed by PointFlow, which only provides point clouds and not SDF, we re-implemented these methods to compare with our approach. It should be noted that since our dataset and PointFlow's are based on the different versions of ShapeNet, and different pre-processing methods, the results may slightly differ from those of the original papers.

4.1. Single-Category Generation

Following previous works [41, 77, 87], we evaluate metrics for 3 models trained with each category (*airplane*, *car*, and *chair*) separately. To ensure a fair comparison with point clouds-based baseline methods that used $2,048$ points to compute metrics, we sampled the same number of points from meshes reconstructed by our method.

We show the quantitative comparisons in Table 1. Our method, `SDF-Diffusion` demonstrated competitive performance in all categories and outperformed other baselines, especially in the 1-NNA (EMD) metric, which is the most important metric designed to overcome the limitations of MMD and COV [77]. In addition, EMD is a more distinctive metric [42] because CD tends to fail to compare point clouds with uneven density, and uniform point cloud density is important to represent 3D shapes well. Since `SDF-Diffusion` reconstructs continuous meshes first and then samples point clouds from the meshes, the point clouds can be evenly arranged, unlike other methods that generate point clouds directly.

Figure 3 shows the qualitative comparisons with the

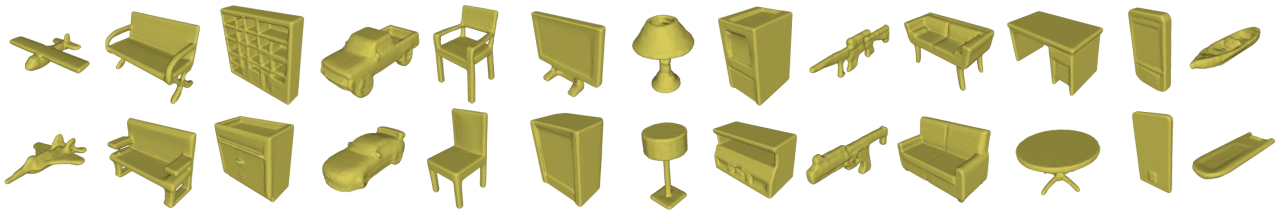

Figure 4. **Generated samples by `SDF-Diffusion` on 13-category classes of the ShapeNet dataset**.

Table 1. **Quantitative evaluation on the ShapeNet dataset about MMD, COV, 1-NNA about 3-category**. MMD, 1-NNA scores are the lower, the better, and the higher COV, the better. MMD-CD and MMD-EMD scores are multiplied by 10^3 and 10^2.

		Trainset	PointFlow [77]	DPM [41]	PVD [87]	ours
Airplane	MMD (CD)	1.51	2.43	**2.24**	2.46	2.37
	COV (CD)	56.93	**50.50**	50.00	45.30	50.25
	1-NNA (CD)	45.92	72.28	67.45	62.62	**56.56**
	MMD (EMD)	2.33	1.67	1.64	1.55	**1.49**
	COV (EMD)	57.43	52.97	52.23	53.96	**55.20**
	1-NNA (EMD)	47.40	62.50	63.37	52.72	**48.14**
Car	MMD (CD)	2.44	2.61	2.57	**2.48**	**2.48**
	COV (CD)	54.07	41.92	44.19	44.33	**47.26**
	1-NNA (CD)	49.47	74.50	75.57	58.48	**58.28**
	MMD (EMD)	1.28	1.39	1.39	1.30	**1.28**
	COV (EMD)	52.34	41.92	41.92	48.33	**52.47**
	1-NNA (EMD)	50.53	71.90	71.90	**51.13**	53.20
Chair	MMD (CD)	8.05	8.27	**7.65**	7.87	8.00
	COV (CD)	54.95	46.53	47.86	48.89	**49.78**
	1-NNA (CD)	52.88	70.83	66.40	55.61	**53.69**
	MMD (EMD)	3.57	4.21	4.08	**3.56**	3.61
	COV (EMD)	53.47	49.63	41.65	**50.37**	49.31
	1-NNA (EMD)	48.74	74.74	76.66	53.03	**51.77**

baseline methods. `SDF-Diffusion` generates plausible 3D shapes like the other approaches; especially, we can directly produce continuous 3D meshes.

4.2. Multi-Category Generation

In addition to single-category, our `SDF-Diffusion` is seamlessly extended to multi-category generation using category information. We train and compare our `SDF-Diffusion` on the 13 categories of the ShapeNet dataset: *airplane*, *bench*, *cabinet*, *car*, *chair*, *display*, *lamp*, *loudspeaker*, *rifle*, *sofa*, *table*, *telephone*, and *vessel*, by conditioning on the category information c. As shown in Fig. 4, `SDF-Diffusion` generates multi-category detailed 3D shapes with high-quality and diversity. Furthermore, we can perform the second stage (*i.e.*, super-resolution) multiple times to generate much higher-resolution samples. Figure 5 illustrates the 3D shapes generated using multiple super-resolution stages, allowing us to achieve 128^3 resolution SDF voxels.

To assess the generation quality of multi-category generation, we compare our model with PVD, which is a diffusion-based state-of-the-art 3D shape generation method. However, PVD and other baselines are designed for only single-category generation. So, we compare ours with the modified version of PVD (PVD* in short) that can receive a category condition with MLPs with Embedding layers similar to ours but without AdaGN to preserve the timestep embedding method of the original PVD. Quantitative results are shown in Table 2. The results of PVD* and `SDF-Diffusion` are trained equally for $1,000$ epochs for only multi-category case. `SDF-Diffusion` shows better results in most categories about 1-NNA metric, especially outperforming EMD-based metrics. Additional qualitative comparisons on the mesh domain will be discussed in the supplementary material.

Table 2. **Quantitative evaluation on the ShapeNet dataset through 1-NNA about CD and EMD with 13-category**. PVD* denotes the modified version of PVD for multi-category.

Category	1-NNA (CD)			1-NNA (EMD)		
	PVD*	ours	trainset	PVD*	ours	trainset
Airplane	**67.20**	70.79	45.92	**57.92**	58.04	47.40
Bench	62.43	**60.22**	53.31	59.67	**57.46**	54.14
Cabinet	**62.10**	62.74	47.13	68.15	**63.06**	50.96
Car	**64.55**	66.29	49.47	57.89	**55.81**	50.53
Chair	**61.89**	63.23	52.88	59.40	**59.33**	48.74
Display	61.47	**44.50**	51.83	58.26	**53.21**	52.29
Lamp	58.87	**58.01**	49.13	61.04	**56.28**	51.95
Loudspeaker	59.32	**57.14**	48.14	57.45	**54.76**	52.17
Rifle	61.39	**60.54**	50.00	58.44	**53.59**	49.79
Sofa	58.04	**52.84**	51.58	54.57	**50.63**	47.32
Table	62.41	**55.76**	49.71	61.82	**56.35**	51.00
Telephone	60.00	**57.14**	48.57	55.71	**54.76**	50.95
Vessel	**59.59**	59.84	51.55	58.29	**54.15**	50.78
Average	61.48	**59.16**	49.94	59.12	**55.96**	50.62

4.3. Application: Shape Completion

We show the applicability of our `SDF-Diffusion` on another task: 3D shape completion with voxel condition. Without modification of network architecture, we show that our method can be applied to 3D shape completion. Specifically, the second stage of our method takes a voxel-shaped condition by concatenating the conditional input (*i.e.*, low-resolution SDF generated by the first stage) with noisy data to denoise. In other words, let the SDF voxel be a single-channel, then we can consider the second stage of our model takes two-channel input and returns a single-channel output. Therefore, using the same network architecture, we can seamlessly train a shape completion model by giving a partial SDF voxel instead of a low-resolution condition.

For partial 3D shapes, we perform a 3D shape comple-

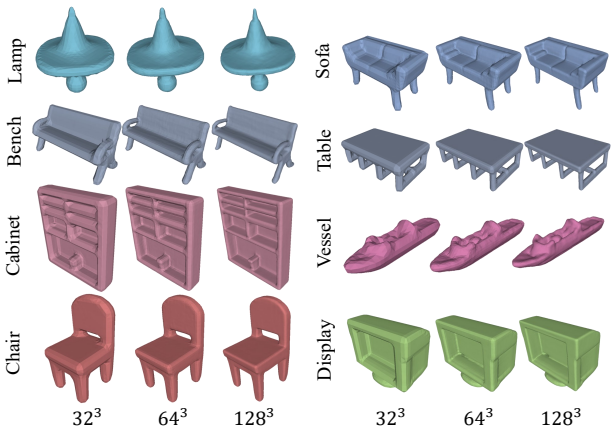

Figure 5. **Visualization of generated 3D shapes at different resolutions**. The first stage model generates coarse shapes in the resolution of 32^3. By performing super-resolution using our second stage model multiple times, we can obtain improved shape detail of 3D shapes with the voxel resolutions of 64^3 and 128^3.

tion based on SDF-Diffision, as shown in Fig. 6. As you can see, we can generate various 3D shapes from given partial 3D shapes, which shows the applicability and flexibility of the proposed SDF-Diffision.

In addition, we also demonstrate the generality of our SDF-Diffusion by using it for the task of 3D shape completion with a partially observed point cloud as a condition. To generate the partial point cloud input, we randomly sample 200 points from the preprocessed ShapeNet dataset using the same method employed in [82]. For out-of-distribution data, we acquire partial point clouds from depth images of the Redwood 3DScans dataset [9]. As the point cloud has a different format than the voxel feature generated by our SDF-Diffusion, we incorporate an additional layer to extract and voxelize the point cloud. Specifically, we employ the PointNet layer [55] from the encoder of the convolutional occupancy network [53], which transforms the point cloud feature into a voxel and gives this feature to each layer of SDF-Diffusion along with the AdaGN. The completed 3D shapes are visualized in Fig. 7. We show SDF-Diffusion can successfully apply unseen out-of-distribution data.

5. Conclusion

In this work, we have proposed SDF-Diffusion, a novel diffusion-based SDF generation approach for 3D shapes. SDF-Diffusion composed of two-stage generates low-resolution SDF and then produces high-resolution SDF conditioned on the low-resolution one. For memory efficiency, we learn the second stage of SDF-Diffusion (*i.e.*, super-resolution) in a patch-wise manner, which allows us to perform super-resolution several times in high-resolution. In addition, SDF-Diffusion can be seamlessly extended to multi-category 3D shape generation and 3D shape completion. Experiments demonstrate that the proposed method can generate high-quality 3D shapes in the SDF domain with superior generative power.

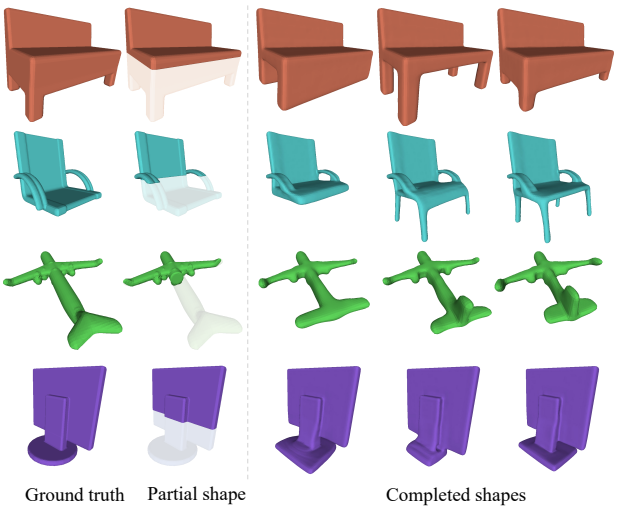

Figure 6. **Generated 3D shapes by 3D shape completion.** SDF-Diffusion can be easily expanded to a 3D shape completion task by modifying only the input head of the network architecture.

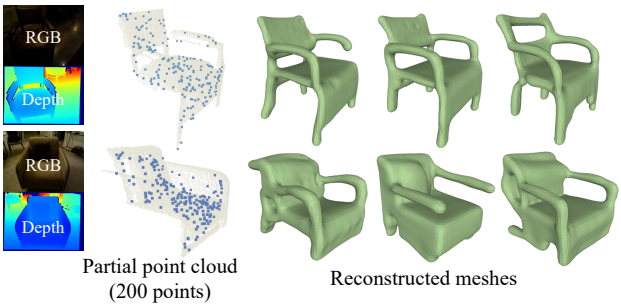

Figure 7. **Generated 3D shapes by 3D shape completion on out-of-distribution data.** SDF-Diffusion shows various completion candidates from out-of-distribution partial point clouds.

Acknowledgments. This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-00612, Geometric and Physical Commonsense Reasoning based Behavior Intelligence for Embodied AI, No.2022-0-00907, Development of AI Bots Collaboration Platform and Self-organizing AI and No.2020-0-01336, Artificial Intelligence Graduate School Program (UNIST)) and Artificial intelligence industrial convergence cluster development project funded by the Ministry of Science and ICT (MSIT, Korea) & Gwangju Metropolitan City.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3D point clouds. In *ICML*, 2018. 1, 2

[2] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Cláudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE transactions on visualization and computer graphics*, 5(4):349–359, 1999. 2

[3] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 6

[4] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. *arXiv preprint arXiv:2009.00713*, 2020. 4

[5] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. In *ICLR*, 2021. 1

[6] Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational lossy autoencoder. *arXiv preprint arXiv:1611.02731*, 2016. 3

[7] Zhiqin Chen, Vladimir G. Kim, Matthew Fisher, Noam Aigerman, Hao Zhang, and Siddhartha Chaudhuri. Decor-gan: 3d shape detailization by conditional refinement. In *CVPR*, 2021. 1, 2

[8] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019. 1, 2

[9] Sungjoon Choi, Qian-Yi Zhou, Stephen Miller, and Vladlen Koltun. A large dataset of object scans. *arXiv preprint arXiv:1602.02481*, 2016. 8

[10] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *NeurIPS*, 2021. 1, 3, 5

[11] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016. 3

[12] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2021. 2

[13] Lin Gao, Tong Wu, Yu-Jie Yuan, Ming-Xian Lin, Yu-Kun Lai, and Hao Zhang. Tm-net: Deep generative networks for textured meshes. *ACM Trans. Graph.*, 2021. 1, 2

[14] Rohit Girdhar, David F. Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *ECCV*, 2016. 2

[15] Christian Häne, Shubham Tulsiani, and Jitendra Malik. Hierarchical surface prediction for 3d object reconstruction. *3DV*, 2017. 2

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020. 1, 3, 4, 6

[17] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *JMLR*, 2022. 3

[18] Wenlong Huang, Brian Lai, Weijian Xu, and Zhuowen Tu. 3d volumetric modeling with introspective neural networks. *AAAI*, 33, 2019. 1

[19] Moritz Ibing, Gregor Kobsik, and Leif Kobbelt. Octree transformer: Autoregressive 3d shape generation on hierarchically structured sequences. *CoRR*, 2021. 1, 2

[20] Moritz Ibing, Isaak Lim, and Leif Kobbelt. 3d shape generation with grid-based implicit functions. In *CVPR*, 2021. 1, 2

[21] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, pages 1125–1134, 2017. 3

[22] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Proc. NeurIPS*, 2022. 1

[23] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, pages 8110–8119, 2020. 2, 6

[24] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006. 2

[25] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM TOG*, 2013. 2

[26] Hyeongju Kim, Hyeonseung Lee, Woo Hyun Kang, Joun Yeop Lee, and Nam Soo Kim. Softflow: Probabilistic framework for normalizing flow on manifolds. In *NeurIPS*, 2020. 1, 2

[27] Jinwoo Kim, Jaehoon Yoo, Juho Lee, and Seunghoon Hong. Setvae: Learning hierarchical composition for generative modeling of set-structured data. In *CVPR*, 2021. 1, 2

[28] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[29] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2013. 3

[30] Roman Klokov, Edmond Boyer, and Jakob Verbeek. Discrete point flow networks for efficient point cloud generation. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *ECCV*, 2020. 1, 2

[31] Wei-Jan Ko, Hui-Yu Huang, Yu-Liang Kuo, Chen-Yi Chiu, Li-Heng Wang, and Wei-Chen Chiu. RPG: learning recursive point cloud generation. *CoRR*, 2021. 1, 2

[32] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. In *ICLR*, 2021. 1

[33] Chun-Liang Li, Manzil Zaheer, Yang Zhang, Barnabás Póczos, and Ruslan Salakhutdinov. Point cloud GAN. *CoRR*, 2018. 2

[34] Yiyi Liao, Katja Schwarz, Lars M. Mescheder, and Andreas Geiger. Towards unsupervised learning of generative models for 3d controllable image synthesis. In *CVPR*, 2022. 2

[35] Or Litany, Alex Bronstein, Michael Bronstein, and Ameesh Makadia. Deformable shape completion with graph convolutional autoencoders. In *CVPR*, 2018. 1

[36] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. *NeurIPS*, 2019. 3

[37] David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. *arXiv preprint arXiv:1610.06545*, 2016. 6

[38] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM SIGGRAPH*, 1987. 2, 3

[39] Troy Luhman and Eric Luhman. Improving diffusion model efficiency through patching. *arXiv preprint arXiv:2207.04316*, 2022. 1

[40] Andrew Luo, Tianqin Li, Wen-Hao Zhang, and Tai Sing Lee. Surfgen: Adversarial 3d shape synthesis with explicit surface discriminators. In *ICCV*, 2021. 1

[41] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *CVPR*, 2021. 1, 3, 6, 7

[42] Zhaoyang Lyu, Zhifeng Kong, XU Xudong, Liang Pan, and Dahua Lin. A conditional point diffusion-refinement paradigm for 3d point cloud completion. In *ICLR*, 2021. 3, 6

[43] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, 2019. 2

[44] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *ECCV*, 2020. 2

[45] Paritosh Mittal, Yen-Chi Cheng, Maneesh Singh, and Shubham Tulsiani. Autosdf: Shape priors for 3d completion, reconstruction and generation. In *CVPR*, 2022. 1, 2

[46] Charlie Nash, Yaroslav Ganin, S. M. Ali Eslami, and Peter Battaglia. PolyGen: An autoregressive generative model of 3D meshes. In *ICML*, 2020. 1, 2

[47] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *ICML*, 2021. 1, 3

[48] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *CVPR*, 2020. 2

[49] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *ICCV*, 2021. 2

[50] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 2

[51] Dario Pavllo, Jonas Kohler, Thomas Hofmann, and Aurélien Lucchi. Learning generative models of textured 3d meshes from real-world images. In *CVPR*, 2021. 1, 2

[52] Songyou Peng, Chiyu Jiang, Yiyi Liao, Michael Niemeyer, Marc Pollefeys, and Andreas Geiger. Shape as points: A differentiable poisson solver. In *NeurIPS*, 2021. 2, 3

[53] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *ECCV*, 2020. 6, 8

[54] Vadim Popov, Ivan Vovk, Vladimir Gogoryan, Tasnima Sadekova, and Mikhail A. Kudinov. Grad-tts: A diffusion probabilistic model for text-to-speech. In *ICML*, 2021. 1

[55] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 2, 3, 8

[56] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 2

[57] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Swish: a self-gated activation function. *arXiv preprint arXiv:1710.05941*, 2017. 4

[58] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 3

[59] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *IEEE TPAMI*, 2022. 3, 4, 5

[60] 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 *CVPR*, 2022. 1, 2

[61] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. In *NeurIPS*, 2021. 1

[62] Dong Wook Shu, Sung Woo Park, and Junseok Kwon. 3d point cloud generative adversarial network based on tree structured graph convolutions. In *CVPR*, 2019. 1, 2

[63] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015. 1, 3, 4

[64] Christiane Sommer, Lu Sang, David Schubert, and Daniel Cremers. Gradient-sdf: A semi-implicit surface representation for 3d reconstruction. In *CVPR*, 2022. 2

[65] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 6

[66] Yongbin Sun, Yue Wang, Ziwei Liu, Joshua E Siegel, and Sanjay E Sarma. Pointgrow: Autoregressively learned point cloud generation with self-attention. In *WACV*, 2020. 2

[67] Qingyang Tan, Lin Gao, Yu-Kun Lai, and Shihong Xia. Variational autoencoders for deforming 3d mesh models. In *CVPR*, 2018. 1, 2

[68] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017. 4, 5

[69] Peng-Shuai Wang, Yang Liu, and Xin Tong. Dual octree graph networks for learning adaptive volumetric shape representations. *ACM TOG*, 2022. 6

[70] Peng-Shuai Wang, Yang Liu, and Xin Tong. Mesh2sdf. https://github.com/wang-ps/mesh2sdf, 2022. 6

[71] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *NeurIPS*, 2016. 1, 2

[72] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, 2018. 5

[73] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015. 2

[74] Jianwen Xie, Yifei Xu, Zilong Zheng, Song-Chun Zhu, and Ying Nian Wu. Generative pointnet: Deep energy-based learning on unordered point sets for 3d generation, reconstruction and classification. In *CVPR*, 2021. 1

[75] Jianwen Xie, Zilong Zheng, Ruiqi Gao, Wenguan Wang, Song-Chun Zhu, and Ying Nian Wu. Learning descriptor networks for 3d shape synthesis and analysis. In *CVPR*, 2018. 1

[76] Xingguang Yan, Liqiang Lin, Niloy J Mitra, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Shapeformer: Transformer-based shape completion via sparse representation. In *CVPR*, 2022. 1, 2

[77] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *ICCV*, 2019. 1, 2, 6, 7

[78] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. In *NeurIPS*, 2020. 2

[79] Kangxue Yin, Zhiqin Chen, Hui Huang, Daniel Cohen-Or, and Hao Zhang. Logan: Unpaired shape transform in latent overcomplete space. *ACM TOG*, 2019. 2

[80] Kangxue Yin, Jun Gao, Maria Shugrina, Sameh Khamis, and Sanja Fidler. 3dstylenet: Creating 3d shapes with geometric and texture style variations. In *ICCV*, 2021. 1

[81] Kangxue Yin, Hui Huang, Daniel Cohen-Or, and Hao Zhang. P2p-net: Bidirectional point displacement net for shape transform. *ACM TOG*, 2018. 2

[82] Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In *ICCV*, 2021. 8

[83] 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 *NeurIPS*, 2022. 1, 3

[84] Dongsu Zhang, Changwoon Choi, Jeonghwan Kim, and Young Min Kim. Learning to generate 3d shapes with generative cellular automata. In *ICLR*, 2021. 1

[85] Yuxuan Zhang, Wenzheng Chen, Huan Ling, Jun Gao, Yinan Zhang, Antonio Torralba, and Sanja Fidler. Image gans meet differentiable rendering for inverse graphics and interpretable 3d neural rendering. In *ICLR*, 2021. 1, 2

[86] X Zheng, Yang Liu, P Wang, and Xin Tong. Sdf-stylegan: Implicit sdf-based stylegan for 3d shape generation. In *CGF*, 2022. 2

[87] Linqi Zhou, Yilun Du, and Jiajun Wu. 3D shape generation and completion through point-voxel diffusion. In *CVPR*, 2021. 1, 3, 6, 7