Point Cloud Synthesis with Stochastic Differential Equations

Tingting Li
Bournemouth University
tli@bournemouth.ac.uk

Meili Wang
Northwest A&F University
wml@nwsuaf.edu.cn

Xiaoxiao Li
Bournemouth University
xliu@bournemouth.ac.uk

Hui Liang
Zhengzhou University of Light Industry
hliang@zzuli.edu.cn

Jian Chang
Bournemouth University
jchang@bournemouth.ac.uk

Jian Jun Zhang
Bournemouth University
jzhang@bournemouth.ac.uk

Abstract

In this paper, we propose a point cloud synthesis method based on stochastic differential equations (SDEs). We view the point cloud generation process as smoothly transforming from a known prior distribution toward the high-likelihood shape by point-level denoising. We introduce a conditional corrector sampler to improve the quality of point clouds. By leveraging Markov Chain Monte Carlo (MCMC) sample, our method can synthesize realistic point clouds. We additionally prove that our approach can be trained in an auto-encoding fashion and reconstruct point cloud faithfully. Furthermore, our model can be extend on a downstream application of point clouds completion. Experimental results demonstrate the effectiveness and efficiency of our method.

Keywords: point cloud synthesis, stochastic differential equations, point cloud reconstruction

1 Introduction

Point clouds is one of the most popular 3D shape representations that can represent diverse shapes with a set of sparse and discrete 3D points. With recent advances in 3D sensors, numerous 3D processing and understanding works based on point cloud have been made [1], such as object classification, semantic segmentation etc. Among these works, deep learning based approaches show superior performance cross various tasks. The past decade have witnessed great advance in the deep neural learning on 3D point cloud [2, 3]. These network require large-scale dataset of point clouds shape to optimize massive parameters [4]. Therefore, create high-quality and expressive point cloud becomes a bottleneck in achieving a powerful neural network.

Recent development in generative models provides an alternative solution for acquiring 3D data, such as variational auto-encoders (VAEs) [5, 6], generative adversarial networks (GANs) [7–9], auto-regressive [10], etc. All aforementioned point cloud synthesis methods are supervised by either the Chamfer Distance (CD) or Earth Mover‘s Distance (EMD) to optimize the network to generate expressive point clouds [11]. However, CD loss does not compute the matching between ground truth and synthesis and emphasizes accuracy rather than uniform distribution of shape. EMD loss can be computed as the minimal value of a linear program, which is sensitive to overall distribution of point cloud, but its computational expense is high.

Diffusion Probabilistic Generative Models [12] emerged as a promising class of generative models and brought state-of-the-art performance on multiple tasks. The training objective of diffusion probabilistic generative models implicitly computes scores at each noise scale, which uses a simple mean squared error (MSE) loss function for training [12]. Therefore, point cloud synthesis based on diffusion probabilistic model can avoid the weakness of previous methods.

The key kernel of diffusion probabilistic generative models is to consecutively move each point from a prior analytical distribution into complex data distribution [12–16]. Point cloud synthesis based on DDPMs [11, 17] involve sequentially corrupting ground truth point cloud with slowly increasing noise, and then a neural network is used to learn to reverse this corruption. However, previous works employ a fixed-step Markov chain to approximate the diffusion process and apply a fixed time-step in the reverse process. This can limit the expression of point clouds synthesized by the network, such as lack of smooth surfaces, or lack of sharp edges.

Inspired by [16], we propose the point cloud synthesis based on SDEs. The overview of key idea is given in Fig 1. Our goal is to learn a transition kernel that can synthesize plausible point clouds based on SDEs. We adopt time embedding to exploit the arbitrary sample of time. Point cloud synthesis can be benefit from this in two aspects: a) the network can better understand the time variable; b) the transform process is smoother and flexible. For example, we can adopt different sample time steps for training phase and inference phase, in which the reverse process can be conducted in arbitrary time length. Increasing sample times can enhance the quality of synthesized point cloud without training the network again. As the purpose SDEs solver is to integrate the reverse-time SDEs for sampling, the SDEs solver alone is not enough to generate high quality point clouds. To this end, we employ normalizing flows formulation to parameterize the shape distribution to prior distribution. Additionally, we introduce Langevin MCMC [18] samplers with SDEs-based approaches to improve over simple progression sampling methods. We apply our model to point cloud generation and unsupervised representation learning. Experimental results demonstrate that our model achieves competitive performance on two learning tasks: point cloud synthesis, that generate point cloud from a global latent variable, and auto-encoding generation.

To summarize, the main contributions of this work include:

1. We propose an improved diffusion probabilistic generative model for point clouds, inspired by stochastic differential equation in generative model. It allows the reverse process can be applied with arbitrary time length. The model can generate high-quality point clouds by flexible and smooth transform.

2. Our method scales adopts randomly sample time-step and thus can be efficiently applied to high-quality point clouds generation.

3. Experimental results show that our method can achieve faithful generation and reconstruction. Besides quantitative comparison, we visualize results for different set-up experiments. Our code is publicly available at git

2 Related Work

Point cloud generative models. The past decade have witnessed great advance in the 3D shape generation with neural networks. Generally speaking, this task relates to many fields in computer graphics and computer vision, such as 3D shape reconstruction based on 2D image [19], depth map generation [20] and 3D shape transform [21]. In this section, we focus on 3D point cloud generation methods, specifically in the area of synthesizing diverse and high-fidelity 3D shapes, which are synthesized point cloud data that are real world examples Roughly speaking, previous works can be divided into four categories based on their learning manner: Auto-encoder based geneation [5, 6], Autogressive-based generation [10], GANs-based generation (generative adversarial networks) [7–9, 22–24] and flow-based generation [17, 19, 21, 25, 26].

The success of diffusion probabilistic distribution approach has inspired many follow-up

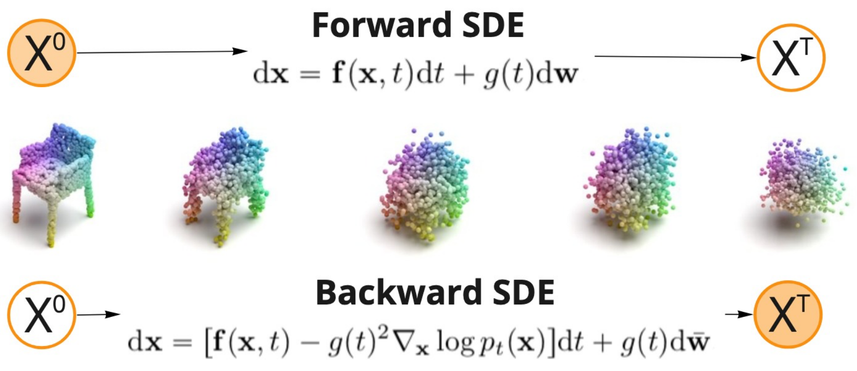
Forward SDE
$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$
X^0
X^T
Backward SDE
$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g(t)^2\nabla_{\mathbf{x}} \log p_t(\mathbf{x})]\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$
X^0
X^T

Figure 1: Overview of point cloud generative modeling through SDEs. We can transform data to a simple noise distribution with a continuous-time SDE and reverse this process for point cloud generation. The reverse-time process can be obtained by the generative model SDEs-Net.

works that extend the diffusion probabilistic approach to 3D point cloud generation. They have viewed 3D point clouds generation as probabilistic distribution transform and exploited auto-encoding architecture with diffusion probabilistic generation approach. For example, Yang et al. [27] proposed to generate 3D point cloud from a standard 3D Gaussian prior, which leverage discrete normalizing flow with affine coupling layers. Cai et al. [21] learned the gradient of the log probability density with respect to point clouds and samples point clouds using Langevin dynamics. Luo et al. [17] exploited the Denoising Diffusion Probabilistic Models (DDPM) with fixed time-steps for point cloud generation. Flowing their work, many researches applied DDPM to various point cloud related-fields, such as point-voxel generation [19], point cloud completion [11] and achieved remarkable results.

Our model is different from these models in that we do not need fixed sample time steps of diffusion probabilistic model, so that the point clouds generation have flexible sample process. In this way, the transform process is smoother and flexible. The reverse process can be conducted in arbitrary time length. In this way, the generated point clouds will have higher quality and sample effectiveness.

Diffusion probabilistic generative models. As describe in Sec.1, most of generative models, such as GAN, VAE, auto-regressive models and flow-based moels rely on delicately design the loss function. Among them, Denoising Diffusion Probabilistic Model (DDPM) [28] is a representative one. It designs a bi-direction process, which is to systematically and gradually destroy data distribution by injecting noise. Then the network learns the reverse diffusion process, yielding a highly flexible and tractable generative model of the data.

Song and Ermon [13] first proposed learning generative models to estimate the gradient of the log probability density through a multi-scale denoising score. After Ho et al. [12] proved the potential of DDPM in term of image generation. Diffusion models are really taking off and emerging as the go-to model for many tasks that requires producing perceptual signals. Ho et al. [12] proposed to use a fixed steps Markov chain to simulate the diffusion process. However, limited by this setup, the sample process is rigid and slow. Therefore, many works focus on the sampling efficient and proposed improved approaches [14, 15, 29]. Because the existing methods do not have strict learning paradigm for diffusion probabilistic models, Song et al. [16] proposed to formulate the diffusion probabilistic models by SDEs and lucidly explained from perspective of variance trend. Our work builds on the theory of [13] and we explore into 3D domain, which is challenging and fundamentally different from aforementioned methods. Different from previous, we leveraging the SDEs to make the transform process of point cloud smoother and flexible. We improved the sampling process to improve the generated point cloud.

3 Method

Given N points in a point cloud $X = \{x_i | i = 1, .., N\} \in \mathbb{R}^{N\times 3}$, we assume p_{data} to be the distribution of the each point cloud X in the dataset. For each point x_i in point cloud X, the status in diffusion process $\{x_i^t\}_{t=0}^T$ can be indexed by a continuous time variable $t \in [0, T]$. The diffusion process is irrelevant of the start distribution of point cloud and the ultimate distribution of the point cloud at $t = T$ is denoted by $p_{latent} = \mathcal{N}(0, \mathbf{I})$, where $\mathcal{N}$ is the Gaussian distribution Formally, let $x^0 \sim p_{data}$ and $x^T \sim p_{latent}$.

Our goal is to learn and generate shape from the distribution of each shape p_{data}. We propose to use stochastic differential equations to model the distribution of the point cloud, which induces a desired shape of points through transforming a prior distribution p_{data}. However, learning a generative model of point clouds directly from an unordered and discrete point cloud is slow and difficult to optimize. As a result, we need to characterize a distribution of the set $\{p_{data}\}$. We parametrize a latent variable $\mathbf{z}$ with continuous normalizing flow [30] that represents the global shape of point cloud. In this case, generating a shape can be represented by a conditional reverse diffusion process.

Sampling points from the induced latent variable guarantees the integrity of point clouds. Nevertheless, the conditional transform process can only roughly estimate the average data distribution.

3.1 Formulation

This diffusion process can be represented as an It$\hat{o}$ SDE [31]. It is the rule for differentiating a function of a stochastic process. In this paper's setting, it is the process that gradually transform 3D shape into 3D Gaussian noise. It can be formulated as:

$$\mathrm{d}x = \mathbf{f}(x, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w} \tag{1}$$

where $\mathbf{w}$ is the standard Brownian motion (Wiener process); $\mathbf{f}(\cdot, t) : \mathbb{R}^d \to \mathbb{R}^d$ is a drift coeffcient of x^t, and $g(\cdot) : \mathbb{R} \to \mathbb{R}$ the diffusion coefficient of x^t. In this paper, the diffusion coefficient is a $d \times d$ scalar matrix. The SDE has a unique strong solution as long as the coefficients are globally Lipschitz in both state and time [32]. Typically, x^T is an unstructured prior distribution (3D Gaussian Distribution), and x^0 and p_{data} shared the similar distribution.

Following SDE, the forward process of diffusion process can be discreted as:

$$x^{(i+1)} = x^i + \mathbf{f}^i(x^i) + \mathbf{G}^i \epsilon_i, \quad i = 0, 1, ..., T-1 \tag{2}$$

Based on Eq.2 and [33], the reverse-time SDE can be discretized as:

$$\begin{aligned} \mathrm{d}x =& \mathbf{f}(x, t) - \mathbf{G}(t)\mathbf{G}(t)^{\mathrm{T}} \nabla_x \log p_t(x)\mathrm{d}t \\ &+ \mathbf{G}(t)\mathrm{d}\bar{\mathbf{w}} \\ \mathrm{d}x =& [\mathbf{f}(x, t) - g^2(t)\nabla_x \log p_t(x)]\mathrm{d}t \\ &+ g(t)\mathrm{d}\bar{\mathbf{w}} \end{aligned} \tag{3}$$

where $\bar{\mathbf{w}}$ is a Brownian motion in the reverse time direction, and dt represents an infinitesimal negative time step. The reversed SDE can be computed once we know the drift and diffusion coefficients of the forward SDE, as well as the output of $p_t(\mathbf{x})$ for each $t \in [0, T]$.

To estimate the gradient $\nabla_x \log p_t(x)$ we train a SDE-Net model s_{θ^*}, where the θ denotes the parameters of network. In this case, the training objective of SDE-Net model $s_\theta(x, t)$ can be denoted as:

$$\begin{aligned} \theta^* = \underset{\theta}{\operatorname{argmin}}\, &\mathbb{E}_t\{\lambda(t)\mathbb{E}_{(x^0,\mathbf{z})}\mathbb{E}_{(x^t|x^0,\mathbf{z})} \\ &[\|s_\theta(x^t, t, \mathbf{z}) - \nabla_{x^t} \log p_{0t}(x^t|x^0, \mathbf{z})\|^2]\} \end{aligned} \tag{4}$$

When the drift $\mathbf{f}$ and diffusion coefficient g of an SDE are not affine, it can be difficult to compute the $\log p_{0t}(x^t|x^0, \mathbf{z})$ in closed form.

As shown in Eq.3 and Eq.4, x^t is influced by the $\lambda(t)$ and brown motion $\mathrm{d}\mathbf{w}$ where $\mathrm{d}\mathbf{w}$ can be denoted by a random noise. Following [14] and [13], we set $\lambda \propto 1/\mathbb{E}\left[\|\nabla_{x^t} \log p_{0t}(x^t|x^0, \mathbf{z})\|^2\right]$ and simplify the SDE in following formulation:

$$\mathrm{d}x = \sigma^t \mathrm{d}\mathbf{w} \tag{5}$$

where the σ denotes the noise scale; $t \in [0, 1]$. In this case,

$$p_{0t}(x^t|x^0, \mathbf{z}) = \mathcal{N}(x^t; x^0, \frac{(\sigma^{2t} - 1)}{2\log\sigma}\mathbf{I}, \mathbf{z}) \tag{6}$$

where $\mathbf{I}$ denotes the variance of data distribution.

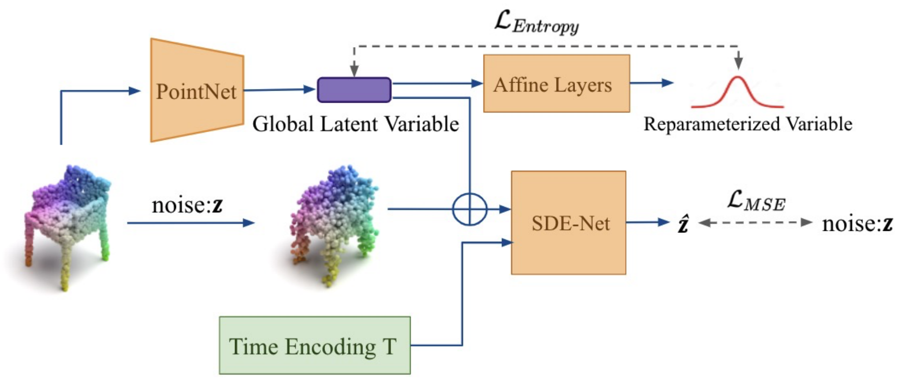

Figure 2: The illustration of training phase of the proposed model.

3.2 Implement Method

Our goal is to generate point clouds with a desired shape, decoded by the latent global representation $\mathbf{z}$. By starting from a noise with the prior distribution p_{latent} and reversing the diffusion process, we are able to obtain a point cloud from the data distribution p_{data}. Crucially, we treat the reverse-time SDE as diffusion process running backwards in a conditional way along with time. Then, the points are sampled from a noise distribution and passed through the reverse diffusion process to form a shape. We define that the θ denotes the diffusion model to approximate stochastic differential equations, and network ϕ learns the distribution of p_{data}. The train phase is show in Fig 2.

3.2.1 Latent Variable Reparamenterization

A normalizing flow [30, 34] is a stack of affine coupling layers $f = \{f_1, .., f_n\}$ as a reversible transform between an prior distribution and a complicated distribution. In particular, $p_{data} = f_n \circ f_{n-1} \circ \ldots f_1(\mathbf{z})$ is the output variable and $\mathbf{z}$ can be estimated from p_{data} via the inverse mapping: $\mathbf{z} = f_1^{-1} \circ \ldots f_n^{-1}(p_{data})$. $\circ$ denotes the Hadamard product. Formally, given the latent variable p_{data} with distribtuion $\mathrm{P}(p_{data})$, let network φ denotes instantiated affine coupling layers that maps p_{data} to the output variable $\mathbf{z}$ with prior distribution $\mathrm{P}(\mathbf{z})$. The exact probability of the output variable is estimated by the change of variables formula:

$$\mathrm{P}(p_{data}) = \mathrm{P}(\mathbf{z}) \left| \det \left(\frac{\partial \varphi}{\partial \mathbf{z}} \right) \right|^{-1} \tag{7}$$

$$, \text{where } \mathbf{z} = \varphi^{-1}(p_{data})$$

3.2.2 Time Embedding

In our diffusion model with stochastic differential equation, the additional input time step t allows a single model to use a common set of parameters to handle different noise levels. However, we find out that the network can ignore the time step t when attach it with input straightforward. Besides, it is suboptimal to increase the parameter of network to handle this parameter, which increases the learning burden. Likewise position embedding in 2D image, we propose to use Gaussian random features [35] to encode t. In particular, the time embedding TE is define as:

$$TE = [\sin(2\pi w t); \cos(2\pi w t)] \tag{8}$$

where operator $[a, b]$ denotes the concatenation; $w \sim \mathcal{N}(0, I)$ is a frozen random matrix.

3.2.3 Training Objective

We implement a point cloud auto-encoder based on the stochastic differential equation in Section 3.1. It is possible to directly apply KL loss over the latent variable outputed by ϕ, but it has been proved that it unavoidably restricts the performance of network [36]. We employ normalizing flow to enhance the representation of network instead of using KL loss to parameterize the latent variable. Formally, leveraging on Eq.4, we rewrite the objective of our network:

$$\begin{aligned}\mathcal{L}(\theta,\phi,\varphi) = &\left(\theta\left(X^t,t,\mathbf{z}\right)\sqrt{\frac{\sigma^{2t}-1}{2\log\sigma}}+\epsilon\right)^2\\ &+D_{KL}\left(\phi(p_{data}|X^0)\|\mathrm{P}(\mathbf{z})\left|\det\left(\frac{\partial\varphi}{\partial\mathbf{z}}\right)\right|^{-1}\right)\\ &+H\left[\phi(p_{data}|X^0)\right]\end{aligned} \tag{9}$$

where $X^t = X^0+\epsilon\sqrt{\frac{\sigma^{2t}-1}{2\log\sigma}}$, and $\epsilon \sim \mathcal{N}(0,I)$ is a random noise. Our model is trained in end-to-end fasion through minimizing the above objective of all point sets in the dataset.

3.2.4 MCMC Sampler

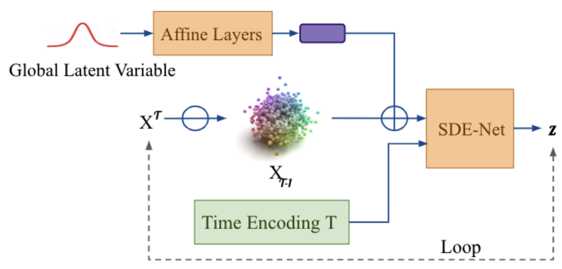

Figure 3: The illustration of inference phase of the proposed model. The dash line represents the loop process.

The inference phase of numerical sampler is shown in Fig 3. It corresponds to the reverse process of SDE. During the inference phase, we build a corrector conditional sampler that combine numerical sampler of reverse process SDE and Langevin MCMC approach. Specifically, we estimate $X^{t-\Delta t}$ for X^t via the SDE-Net model s_θ and global latent variable $\mathbf{z}$ on each time step, and then we use Langevin MCMC sampler to refine X^t. This simplified corrector conditional sampler is as follows Algorithm.1:

As shown in Algorithm.1, r denotes the signal-to-noise ratio of Langvein MCMC sampler. $\|\omega\|_2$ is a random noise with Gaussian distribution. ω, r and output of the SDE-Net s_θ jointly determine the step size ϵ. This additional step that ref as the corrector step, helps us to obtain a more accurate point cloud.

Algorithm 1 Conditional Corrector Smapler

```
Initialization: $\{x^T, \mathbf{z} \sim \mathcal{N}(\mathbf{0},\mathbf{I}), \sigma, r\}$
for $i \leftarrow T$ to $1$ do
    for $j \leftarrow 1$ to $n$ do
        $\omega \sim \mathcal{N}(0,1)$# Sample random noise
        $g \leftarrow s_\theta(X^i_{j-1}, \sigma^i, \mathbf{z})$# Predict gradient
        $\epsilon \leftarrow 2(r\|\omega\|_2/\|g\|_2)^2$# Refine gradient
        $X^i_j \leftarrow X^i_{j-1} + \epsilon g + \sqrt{2\epsilon}\omega$
    $X^{i-1}_0 \leftarrow X^i_n$
```

4 Experiment

We introduce the dataset of our experiment in Sec.4.1, the architecture of our network in Sec.4.2 and common evaluation metrics for point cloud generation task in Sec.4.3. In Sec.4.4, we show some qualitative results and compare the quality of point cloud synthesis based on our method with previous generative models of 3D point clouds in terms of discussed evaluation metrics. In Sec.4.5, we further prove the effectiveness of our model training in auto-encoder fashion and evaluate the representation learning ability. In Sec.4.6, we evaluate our model's performance by point cloud completion.

4.1 Datasts

We carry out point cloud generation experiments on the ShapeNet datasets [37]. It consists of 51,127 point clouds for training set and 1,184 for testing from 55 object categories. We take the main experiment on chair, airplane, car and guitar to prove the effectiveness of our method. The proportion of training, testing and validation sets respectively are 80%, 15% and 5%. We ramdomly sample 2048 points from each of the shape for training. We emphasise that our method is not limited by the sample number of points and it can generate upsample and downsample point clouds as well.

4.2 Implement Details

We adopt PointNet for the architecture of encoder ϕ. As in [30], latent variable reparameterzation φ uses 3 layers with 256 hidden units and Relu activation function. We employ an MLP architecture for modeling network θ with stochastic differential equations. We apply 6 layers of full linear for the model θ. And we apply 1 layer of 64 hidden units for the Time encoding. Models are trained with Adam over 10,000 iterations with a batch size of 128 and an initial learning rate of 10^{-3}. During the inference phase, we set $T = 10^3$ to generate each point cloud.

4.3 Evalution Metrics

For point cloud synthesis, we follow the evaluation set-up in [27] and [23] to compare in terms of minimum matching distance (MMD) [23], the coverage score (COV) [23], 1-NN classifier accuracy (1-NNA) [27] and the Jenson-Shannon divergence (JSD) [27]. Minimum Matching Distance (MMD) is a metric to compute fidelity of synthesized point cloud. The coverage score (COV) is to estimate the generative ability of model, which is whether the generative point cloud cover all the existing mode of the dataset. The 1-NNA is used to measure the 1-NN classifier between our synthesized point cloud and ground truth samples. If the generated shapes like samples from the ground truth distribution, when the score nears to 50%, that means the generated shapes close to real data. The Jenson-Shannon divergence (JSD) computes the similarity between the our synthesized point cloud and the test set of ground truth.

For point cloud auto-encoding, we adopt CD and the EMD to evaluate the reconstruction quality of the point clouds that we introduce on Sec. 1.

4.4 Point Cloud Synthesis

Fig 4 shows some examples of point clouds generated by our model.

The results show that our method can synthesize reasonable point cloud with distinguish structure and clear surface. We normalize each generated shape and evaluate the our generated point clouds by the metrics in Section 4.3. We quantitatively compare our method with the following state-of-the-art generative models: PC-GAN [23], GCN-GAN [38], TreeGAN [9] and PointFlow [27]. The comparison results are shown in Tab 1. It can be seen that our method achieves state-of-art in Chair category and reaches competitive result in another metrics.

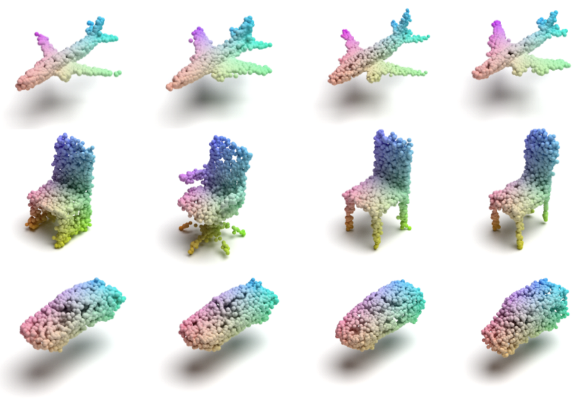

Figure 4: Examples of point clouds synthesized by our model. Each shape is generated from a global latent variable whose distribution is Gaussian distribution

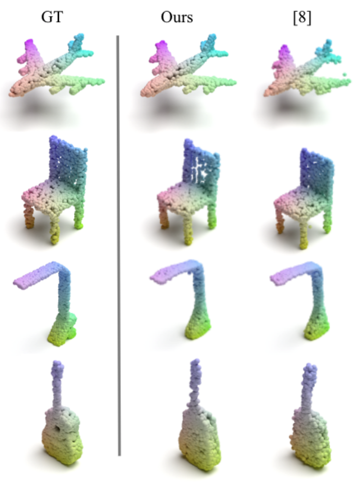

Figure 5: Examples of point clouds reconstruction of auto-encoder.

4.5 Auto-encoder for Point Cloud

We compare with state-of-the-art point cloud auto-encoder: AtlasNet [39], PointFlow [27] and ours. We evaluate the quality of our method with three categories: airplane, chair and whole dataset and the comparison results are shown in Tab 2. As shown in Tab 2, our method achieve better performance in CD score and competitive results when compared in EMD. We emphasize that all the point cloud in comparison experiments contain same number of points and normalized by same approach. We visualize the

Table 1: Comparison of point cloud generation performance.

Category	Model	MMD(↓)		COV(%,↑)		1-NNA(%,↓)		JSD(↓)
		CD	EMD	CD	EMD	CD	EMD	-
Airplane	[7]	3.819	1.810	42.17	13.84	**77.59**	98.52	6.188
	[38]	4.713	1.650	39.04	18.62	89.13	98.60	6.669
	[9]	4.323	1.953	39.37	8.40	83.86	99.67	15.646
	[27]	**3.688**	**1.090**	**44.98**	**44.65**	**66.39**	**69.36**	**2.236**
	Ours	**3.508**	**1.132**	**41.02**	**39.20**	78.00	**82.12**	**2.383**
Chair	[7]	13.436	3.104	46.23	22.14	69.67	100.00	**6.649**
	[38]	15.354	2.213	39.84	35.09	77.86	95.80	21.708
	[9]	14.936	3.613	38.02	6.77	74.92	100.00	13.282
	[27]	13.631	1.856	41.86	43.38	**66.13**	**68.40**	12.474
	Ours	**12.879**	**1.819**	**39.53**	**41.86**	**69.46**	**73.50**	**9.499**

Table 2: Comparison of point cloud reconstruction performance.

Dataset	Metric	[39](S1)	[27]	Ours
Airplane	CD	**2.000**	2.420	2.921
	EMD	4.311	**3.311**	3.624
Chair	CD	5.479	6.795	**6.631**
	EMD	5.550	5.008	**4.578**
All	CD	6.906	7.550	**6.130**
	EMD	5.617	5.172	**4.456**

point clouds reconstruction as shown in Fig 5. It can be seen that our method can reconstruct faithful and clean point clouds.

In addition, we project the global latent variable produced by the encoder of auto-encoder and interpolate between them. We further implement extrapolation and visualize point cloud in Fig 7. The extrapolation results demonstrate that our model is able to learn informative representations.

4.6 Point Cloud Completion

During inference, our model can generate a sequence of point-wise move distance sampling each point cloud, which allows point cloud completion. Motivated by this property, we conduct an additional experiment to point cloud completion as an application for point cloud synthesis. Specifically, we use the partial point cloud as the input and use the pre-trained encoder to estimate the global shape variable. Then we employ the global shape variable and the partial point cloud to synthesize the completed point cloud. The visualized qualitative results are shown in Fig 6. As shown in results, our model can complete precise point cloud when the reference point cloud is sparse.

5 Conclusions

We presented a framework for point cloud generative model based on SDE. Our work brought new point cloud conditional generation approach to the family of point cloud generation based on diffusion models. By leveraging the time encoding and SDE, our method can make the transform between the noise and point cloud more smooth and flexible. This makes diffusion process can be sampled in anytime step without re-train the model which enhance the quality and effectiveness of generated point cloud. Additionally, the we leveraged Lagvien MCMC sample to improve the quality of generated point cloud. Experimental results demonstrated that the proposed model can generate expressive point cloud and achieve competitive results compared with other methods.

References

[1] Yongheng Zhao, Tolga Birdal, Haowen Deng, and Federico Tombari. 3d point capsule networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1009–1018, 2019.

[2] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[3] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets

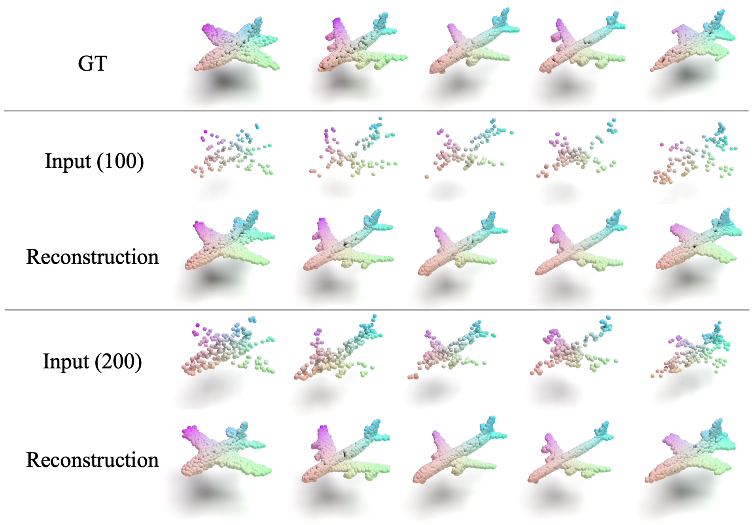

Figure 6: Visualized experimental results of point cloud completion.

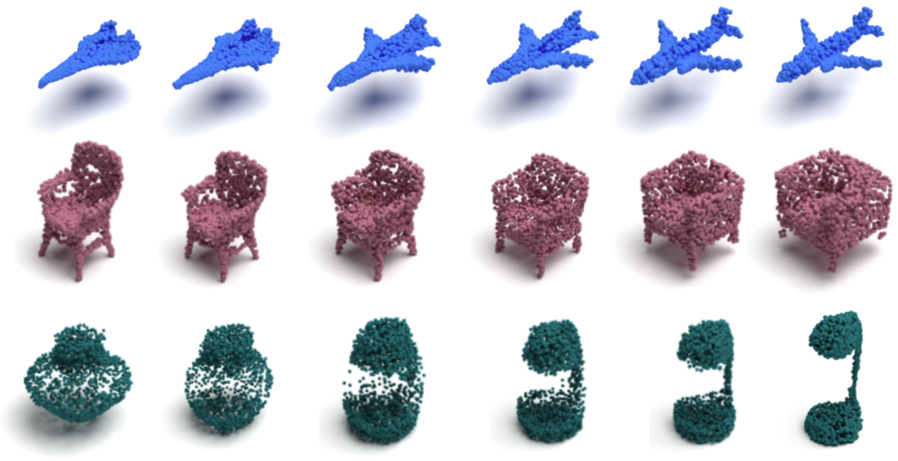

Figure 7: Global latent variable interpolation and extrapolation.

in a metric space. *Advances in neural information processing systems*, 30, 2017.

[4] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16259–16268, 2021.

[5] Matheus Gadelha, Rui Wang, and Subhransu Maji. Multiresolution tree networks for 3d point cloud processing. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 103–118, 2018.

[6] Maciej Zamorski, Maciej Zieba, Piotr Klukowski, Rafał Nowak, Karol Kurach, Wojciech Stokowiec, and Tomasz Trzciński. Adversarial autoencoders for compact representations of 3d point clouds. *Computer Vision and Image Understanding*, 193:102921, 2020.

[7] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *Advances in neural information processing systems*, 29, 2016.

[8] Sameera Ramasinghe, Salman Khan, Nick Barnes, and Stephen Gould. Spectral-gans for high-resolution 3d point-cloud generation. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8169–8176. IEEE, 2020.

[9] Dong Wook Shu, Sung Woo Park, and Junseok Kwon. 3d point cloud generative adversarial network based on tree structured graph convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3859–3868, 2019.

[10] Yongbin Sun, Yue Wang, Ziwei Liu, Joshua Siegel, and Sanjay Sarma. Pointgrow: Autoregressively learned point cloud generation with self-attention. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 61–70, 2020.

[11] Zhaoyang Lyu, Zhifeng Kong, Xudong Xu, Liang Pan, and Dahua Lin. A conditional point diffusion-refinement paradigm for 3d point cloud completion. *arXiv preprint arXiv:2112.03530*, 2021.

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[13] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

[14] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33:12438–12448, 2020.

[15] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[16] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[17] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2837–2845, 2021.

[18] Giorgio Parisi. Correlation functions and computer simulations. *Nuclear Physics B*, 180(3):378–384, 1981.

[19] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5826–5835, 2021.

[20] Kejie Li, Trung Pham, Huangying Zhan, and Ian Reid. Efficient dense point cloud

object reconstruction using deformation vector fields. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 497–513, 2018.

[21] Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. Learning gradient fields for shape generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[22] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[23] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pages 40–49. PMLR, 2018.

[24] Rinon Gal, Amit Bermano, Hao Zhang, and Daniel Cohen-Or. Mrgan: Multi-rooted 3d shape generation with unsupervised part disentanglement. *arXiv preprint arXiv:2007.12944*, 2020.

[25] Roman Klokov, Edmond Boyer, and Jakob Verbeek. Discrete point flow networks for efficient point cloud generation. In *European Conference on Computer Vision*, pages 694–710. Springer, 2020.

[26] Hyeongju Kim, Hyeonseung Lee, Woo Hyun Kang, Joun Yeop Lee, and Nam Soo Kim. Softflow: Probabilistic framework for normalizing flow on manifolds. *Advances in Neural Information Processing Systems*, 33:16388–16397, 2020.

[27] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4541–4550, 2019.

[28] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[29] Daniel Watson, Jonathan Ho, Mohammad Norouzi, and William Chan. Learning to efficiently sample from diffusion probabilistic models. *arXiv preprint arXiv:2106.03802*, 2021.

[30] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

[31] Kiyosi Itô. Stochastic integration. In *Vector and Operator Valued Measures and Applications*, pages 141–148. Elsevier, 1973.

[32] Bernt Øksendal. Stochastic differential equations. In *Stochastic differential equations*, pages 65–84. Springer, 2003.

[33] Brian DO Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

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

[35] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33:7537–7547, 2020.

[36] 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.

[37] 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.

[38] Diego Valsesia, Giulia Fracastoro, and Enrico Magli. Learning localized generative models for 3d point clouds via graph convolution. In *International conference on learning representations*, 2018.

[39] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 216–224, 2018.