The Visual Computer manuscript No.
(will be inserted by the editor)

Diff-PCG: Diffusion Point Cloud Generation Conditioned on Continuous Normalizing Flow

Ting Yu · Weiliang Meng · Zhongqi Wu · Jianwei Guo · Xiaopeng Zhang

Abstract With the continuous advancement of computer technology and graphic capabilities, the creation of 3D point clouds holds great promise across various fields. However, previous methods in this area are still facing huge challenges, such as complex training setups and limited precision in generating high-quality 3D content. Taking inspiration from the denoising diffusion probabilistic model, we propose *Diff-PCG*, a Diffusion Point Cloud Generation Conditioned on Continuous Normalizing Flow for 3D generation. Our approach seamlessly combines forward diffusion and reverse processes to produce high-quality 3D point clouds. Moreover, we include a trainable continuous normalizing flow that controls the foundational structure of the point cloud to enhance the representation ability of the encoded information. Extensive experiments validate the efficacy of our approach in generating high-quality 3D point clouds.

Keywords 3D Shape Generation · Diffusion Model · Continuous Normalizing Flow · Point Cloud

1 Introduction

The rapid progress in computer graphics and artificial intelligence has fueled the growth of 3D content generation in various domains, including game, film, augmented reality, and metaverse. Traditional approaches rely on manual modeling or modification, resulting in high-quality models but requiring extensive manual labor and time investment.

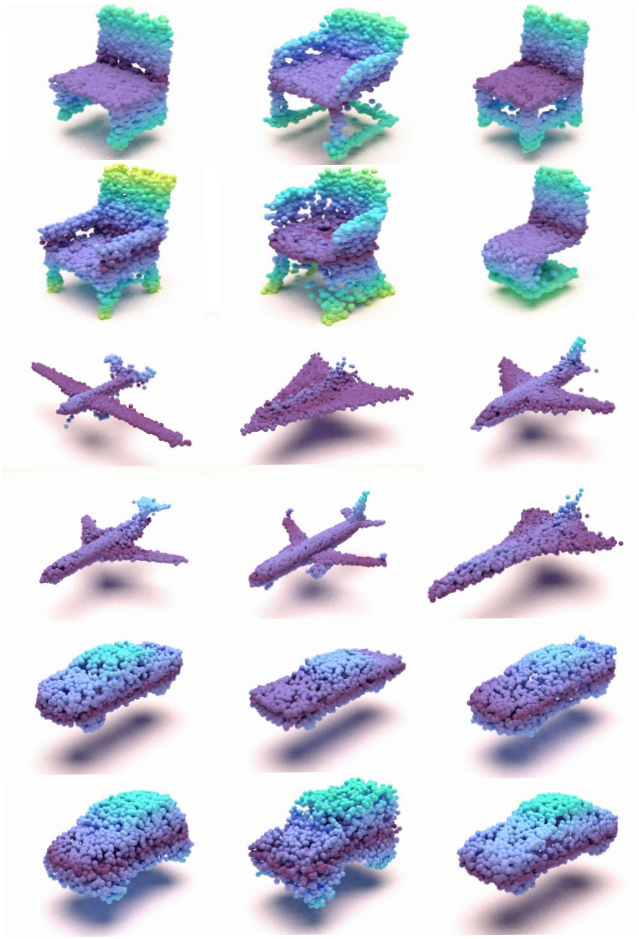


Fig. 1 Examples of 3D shapes generated by our proposed *Diff-PCG*, including chairs, airplanes, and cars. Each sample contains 4096 points, rendered by Mitsuba.

Ting Yu, Weiliang Meng, Jianwei Guo, Xiaopeng Zhang
State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China;
School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China.

Zhongqi Wu
Institute of Science and Development, Chinese Academy of Sciences, Beijing, China.

Jianwei Guo and Xiaopeng Zhang are corresponding authors: jianwei.guo@nlpr.ia.ac.cn; xiaopeng.zhang@ia.ac.cn

However, the emergence of generative modeling for three-dimensional shapes holds immense potential across diverse fields of 3D content creation and has become a thriving research area. 3D generative models currently in use are built on different frameworks, such as generative adversarial networks (GANs) [1–3], 3D CNNs [4], variational autoencoders (VAEs) [5–8], normalizing flows [9, 10], autoregressive models [11, 12], and others [13, 14]. At the same time, unlike the RGB format of 2D images, 3D assets have rich expression structures, including mesh, voxel, point clouds, function, implicit field, etc.

Point clouds have gained popularity as a common 3D representation due to their ability to capture fine details and achieve high resolutions in comparison to voxel grids [4,15]. They act as an important step toward using more advanced representations such as meshes [16] and are known for their capacity to model complex 3D structures with precision [17–19]. Machine learning techniques have shown promise in generating 3D point clouds [20, 21]. However, these approaches face various challenges that require continuous research and development efforts, including the need for extensive training data, long training times, inference durations, and limitations in output quality. Yang et al. [10] proposed a flow-based learning model capable of learning latent features of point cloud shapes, thereby better-guiding point cloud generation based on inverse transformations. However, flow-based models also face challenges such as computational complexity and training difficulties. Luo et al. [22] introduced diffusion models into 3D point cloud generation tasks; however, as mentioned by zhou et al. [23], the performance of PointNet encoders in point cloud reconstruction is not as impressive as PVCNN module [24].

In this paper, we propose Diff-PCG, a point cloud generation framework that integrates continuous normalizing flow modules along with diffusion models. Our method utilizes the power of the diffusion model for generating point clouds while leveraging the continuous normalizing flow to enhance the expressive capacity of the latent variable in capturing shape priors. By building upon the foundational principles of the diffusion model, we employ a forward diffusion process that introduces random coordinate bias to the point cloud, progressively transforming it into a fully disordered state. Subsequently, the point cloud is systematically denoised through the reverse process to restore its original shape. Furthermore, we demonstrate that using a learnable continuous normalizing flow to represent the shape prior of the point cloud expands the range of shape latent variable expressions available.

To summarize, our main goal is to propose a robust, efficient, and accurate point cloud generation algorithm, with the following significant contributions:

- We propose a diffusion-based point cloud generation framework with PVCNN as the encoder, which not only captures point cloud features better than methods based on PointNet but also exhibits greater stability in training compared to previous methods.
- We parameterize the shape latent variable with a Continuous Normalization Flow (CNF), thereby enhancing its ability to express the details of shapes.
- We conduct extensive experiments, validating that our model achieves competitive performance in the task of point cloud generation.

2 Related Work

In this section, we review existing diffusion models and explore using continuous normalization flow to learn shape priors for our point cloud generation. We also discuss alternative deep learning algorithms, emphasizing key challenges and the need for further research to improve precision, efficiency, and versatility in generation models.

Diffusion Model. The diffusion model is a highly effective generative model renowned for its ability to generate realistic images. Initially introduced by Sohl-Dickstein et al. [25] in the realm of physics, this model was later adapted for deep learning-based image generation. However, the original work primarily focused on mathematical derivations and lacked practical code demonstrations, which limited its visibility. Fortunately, subsequent advancements in the field propelled the diffusion generative model into the spotlight. Song et al. [26] introduced the Neural CDE Sampler (NCSN), while Ho et al. [27] from Google Brain presented the Denoising Diffusion Probabilistic Model (DDPM). Additionally, because DDPM parameterizes only the mean of the Gaussian distribution, subsequent improvements by DDIM [28] involve parameterizing both the mean and the variance. These significant contributions built upon the original concept and brought substantial progress by achieving remarkable results. As a result, the diffusion generative model has firmly established itself as a prominent approach in the field.

Point Cloud Generation. Deep learning has made substantial advancements in improving the performance of various tasks related to point clouds, which are sets of 3D points representing the surfaces of objects or scenes. These tasks include classification [29, 30], segmentation [31], and reconstruction [32]. In recent years, there has been remarkable progress in the field, particularly in the area of point cloud synthesis which involves generating or completing point clouds from limited or incomplete data.

Before the breakthrough of the diffusion model, the task of point cloud generation was mainly completed by some previous methods. For instance, variational auto-encoders (VAEs) [33] encode point cloud features into a Gaussian distribution space and then sample from that distribution to recover point cloud data [34, 35]. Deep AutoEncoder [5]

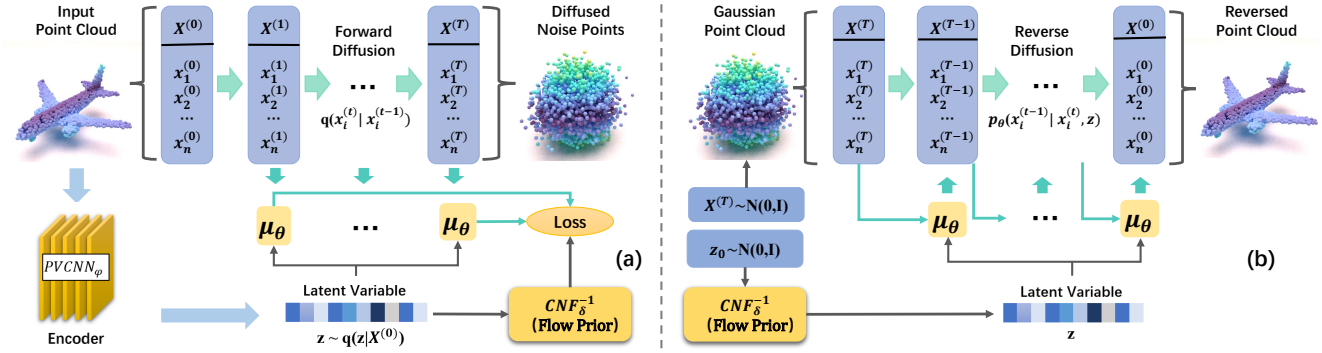


Fig. 2 The architecture of our model is structured as follows: (a) During the training process, we convert the point cloud into a latent variable using the PVCNN module, which acts as a trainable parameter for the neural network of continuous normalizing flow and as an input parameter for the diffusion model. (b) For the generation process, we sample noisy priors and 3D noisy point clouds from normal distributions. Subsequently, a fixed number of reverse diffusion steps are executed to generate the shape.

demonstrates superior geometric data representation, enabling shape manipulation and improved generative models for point clouds. Additionally, generative adversarial networks (GANs) [36] have been explored for both raw data space and latent space generation of point clouds. These methods [3, 37] often employ autoencoders trained using heuristic loss functions such as Chamfer distance (CD) or earth mover's distance (EMD) to measure the dissimilarity between two sets of points. Furthermore, auto-regressive models [11, 38, 39] with a discrete point distribution have been employed to generate point clouds, with a fixed number of points per shape.

Despite these advancements, many existing methods in point cloud synthesis often focus on sampling a specific number of points from the high-dimensional distribution of the point cloud to create the final point cloud data. While this approach has been successful in various applications, further research is needed to explore and refine the generation and completion of point clouds, particularly in scenarios with complex or irregular data distributions. The continuous development of deep learning techniques is expected to drive further progress in this field, enabling more accurate and versatile point cloud synthesis methods. Overall, these approaches have significantly advanced the field of point cloud generation, but there is still ample room for further innovation and improvement.

3 Method

3.1 Overview

The overall flowchart of our algorithm framework can be seen in Fig. 2. As highlighted in the study by Zhou et al. [23], the proficiency of PointNet encoders in reconstructing point clouds falls short compared to the PVCNN module [24]. Therefore, to enhance the performance of the diffusion-based point cloud generation method, during the training phase of our methodology, we employ the PVCNN [24] model as the encoder of our architecture, encoding the input point cloud data into latent variables. These latent variables serve as representations that encapsulate the underlying shape characteristics. Our approach differs from traditional methods by not directly sampling the Gaussian prior for the shape hidden variable from a standard normal distribution. Instead, we implement a process where these latent variables undergo continuous normalized flow during training. This flow significantly shapes the distribution of these variables in a more adaptable way, enhancing the distribution's capacity to handle data intricacies.

Our training framework primarily involves encoding input point clouds into shape latent variables using the PVCNN module. Subsequently, these shape latent variables undergo learning within a continuous normalization flow module and simultaneously serve as input parameters for our neural network during training. Our training loss comprises two main components: one involves the KL divergence between the precomputed Gaussian distribution of point clouds during the forward diffusion process and the Gaussian distribution fitted by the neural network during the reverse diffusion process. The other component pertains to the KL divergence between the Gaussian distribution of the shape latent variables learned from the continuous normalization flow module and the actual Gaussian distribution obtained from the encoder's encoding process.

During the generation phase after training, we sample from a normal distribution to obtain noise prior and 3D noise point cloud data. The noise prior transforms into the shape latent variable through continuous normalization flow. Both transformed shape latent variables and 3D noise point cloud data feed into the trained neural network. Our generated point cloud data exhibit high quality, and achieve compet-

itive quantitative indicators and visualizations compared to previous methods.

Subsequent sections elucidate the principles behind the diffusion model, encompassing forward and reverse diffusion processes. We delve into the continuous normalized flow concept and its role in learning shape priors. Finally, we present the training objective, which involves composing the training loss function.

3.2 Diffusion Model for Point Cloud Generation

The original diffusion model, designed for generating 2D images, has been transformed to make it applicable for the generation of 3D point clouds. Instead of relying on 2D-pixel noise, we have replaced it with a bias introduced into the 3D position coordinates. This modification involves two distinct yet interconnected processes:

(i) **Fixed Forward Diffusion Process (q)**: In this process, a fixed forward diffusion process denoted as "q" comes into play. It gradually introduces random 3D Gaussian noise into the initial point cloud. This noise addition takes place progressively until the point cloud evolves into a state where it consists solely of noise. This marks the initial phase of the adaptation.

Specific to the 3D point cloud data format, let's consider the dataset $q(X^{(0)})$ from which we extract a point cloud ($X^{(0)} = \{x_i^{(0)}\}_{i=1}^N$). This point cloud consists of N points, where each point x_i is drawn from a distribution $q(x_i^{(0)}|z)$, with z representing the latent variable that characterizes the shape of the given point cloud. As mentioned earlier, the forward diffusion process gradually transforms this point cloud into a disordered state. To model this process, we adopt a Markov chain framework:

$$q(x_i^{(1:T)}|x_i^{(0)}) = \prod_{t=1}^{T} q(x_i^{(t)}|x_i^{(t-1)}) \tag{1}$$

The equation above succinctly formulates the Markov diffusion kernel $q(x_i^{(t)}|x_i^{(t-1)})$. Which is rooted in the framework of the original diffusion model [25], and serves as the foundational framework upon which our adaptation for 3D point cloud generation is constructed. By using this kernel, we can precisely control the diffusion process, enabling the structured point cloud to change into a noise-dominated form, and then denoise it to restore the original shape:

$$q(x^{(t)}|x^{(t-1)}) = \mathcal{N}(x^{(t)}|\sqrt{1-\beta_t}x^{(t-1)}, \beta_t I), t \in [1, T] \tag{2}$$

where the β_t parameter is incrementally adjusted in DDPM using a linear strategy. Also, with preconfigured parameters for the forward diffusion process, we utilize the cumulative nature of the Gaussian distribution to derive directly the Gaussian distribution for the forward diffusion at any time step t from the original image $x^{(0)}$. This direct derivation enables obtaining the resulting image $x^{(t)}$ directly:

$$\begin{aligned} q(x^{(t)}|x^{(0)}) &= \mathcal{N}(x^{(t)}; \sqrt{\bar{\alpha}_t}x^{(0)}, (1-\bar{\alpha}_t)I), \\ \alpha_t &:= 1-\beta_t, \;\; \bar{\alpha}_t = \prod_{s=0}^{t} \alpha_s \end{aligned} \tag{3}$$

This allows us to randomly select a time step t from the range $[0, T]$ while using stochastic gradient descent for loss computation, adding flexibility to the training.

Another benefit is the option to adjust the neural network to forecast noise $\epsilon_\theta(x^{(t)}, t)$ at time step t rather than the mean $\mu_\theta(x^{(t)}, t)$ of the Gaussian distribution. Predicting noise is typically simpler than predicting the mean, offering advantages. Their equivalence can be derived as follows:

$$\mu_\theta\left(x^{(t)}, t\right) = \frac{1}{\sqrt{\alpha_t}}\left(x^{(t)} - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta\left(x^{(t)}, t\right)\right) \tag{4}$$

(ii) **Trainable Reverse Denoising Process** (p_θ): Unlike the fixed forward diffusion process, our adaptation's second part involves a trainable reverse denoising process performed by a neural network (p_θ). This network aims to learn and execute the transformation of a noise-only point cloud back into a structured shape. Through gradual denoising, it restores the underlying shape from the noisy point cloud generated during forward diffusion.

If we have access to the inverse sampling distribution $p(x_{t-1}|x_t)$ at each time step t, we can gradually reconstruct the original point cloud from pure noise. However, as this distribution is often unknown, we need to learn its mean and variance through a neural network. In the original DDPM formulation, the variance is predetermined and doesn't need learning. Hence, we use the neural network to approximate the distribution, ensuring an efficient fit:

$$\begin{aligned} p_\theta(x^{(t-1)}|x^{(t)}, z) &= \mathcal{N}(x^{(t-1)}; \mu_\theta(x^{(t)}, t, z), \sigma_\theta(t)), \\ \sigma_\theta(x^{(t)}, t, z) &= \sigma_t^2 I = \beta_t I \end{aligned} \tag{5}$$

then we can formulate the reverse diffusion process at any time step:

$$p_\theta(x^{(0;T)}|z) = p(x^{(T)}) \prod_{t=1}^{i} p_\theta(x^{(t-1)}|x^{(t)}, z) \tag{6}$$

where μ_θ symbolizes the mean of the point cloud's associated distribution, governed by θ, while the latent variable z embodies the point cloud's shape features. For simplicity, we denote the overall point cloud distribution as $X(0)$ in

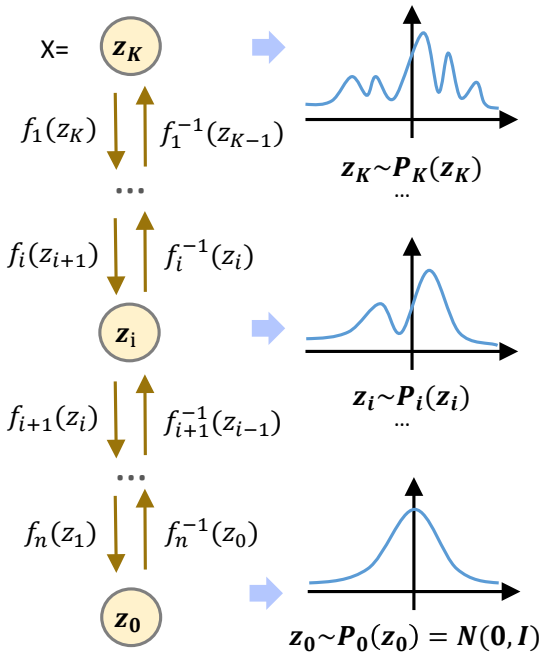


Fig. 3 Normalizing flow aims to model complex probability distributions, allowing the generation of samples that align with the true data distribution from a simple and easily samplable distribution.

subsequent sections. As points within a cloud are independently sampled, the cloud's overall probability is obtained by multiplying probabilities assigned to each point:

$$q(X^{(1:T)}|X^0) = \prod_{i=1}^{n} q(x_i^{(1:T)}|x_i^{(0)}),$$
$$p_\theta(X^{(0:T)}|z) = \prod_{i=1}^{n} p_\theta(x_i^{(0:T)}|z) \tag{7}$$

3.3 Continuous Normalizing Flow for Shape Prior

The original normalizing flow [40] applies invertible mappings $(f_1, ..., f_n)$ to latent variable y from distribution $P(y)$. This transforms an initial distribution into a more complex one, generating output $(x = f_n \circ f_{n-1} \circ ... \circ f_1(y))$. To find the output's probability density, we use the change of variables formula. The inverse flow operation $y = f_1^{-1} \circ ... \circ f_n^{-1}(x)$ computes y from x, as shown in Fig 3. In practical implementations, these mappings are neural networks designed for efficient Jacobian determinant computation.

Chen et al. [41] extended the concept of normalizing flow from a discrete sequence to a continuous transformation. This extension involves defining the transformation f using a continuous-time dynamic equation: $\frac{\partial y(t)}{\partial t} = f(y(t), t)$, where f is a neural network with a flexible architecture. Consequently, the continuous normalizing flow (CNF) model for the distribution $P(x)$, given a prior distribution $P(y)$ at the initial time, can be expressed as follows:

$$x = y(t_0) + \int_{t_0}^{t_1} f(y(t), t)dt, \quad y(t_0) \sim P(y),$$
$$\log P(x) = \log P(y(t_0)) - \int_{t_0}^{t_1} \mathrm{Tr}\left(\frac{\partial f}{\partial y(t)}\right) dt \tag{8}$$

To find the value of y at time t_0 $(y(t_0))$, we use the inverse flow operation: $y(t_0) = x + \int_{t_1}^{t_0} f(y(t), t)dt$. For estimating outputs and input gradients, a black-box ordinary differential equation (ODE) solver is utilized. This solver efficiently approximates the necessary solutions and gradients.

Shape prior based on CNF: The performance of the generative model can be constrained when sampling the latent variable z from a fixed Gaussian prior distribution. To address this limitation, we introduce the continuous normalizing flow (CNF) to parameterize a trainable prior. Mathematically, we redefine the KL divergence term in the following equation to incorporate the CNF-based prior:

$$D_{KL}(Q_\phi(z|x)||P_\delta(z)) = -\mathbb{E}_{Q_\phi(z|x)}[\log P_\delta(z)] - H[Q_\phi(z|X)] \tag{9}$$

where the prior distribution, denoted as $P_\delta(z)$ (δ represents the learnable parameters), is derived by applying a continuous normalizing flow (CNF) to a simple Gaussian distribution $P(w) = N(0, I)$. The entropy term H quantifies the

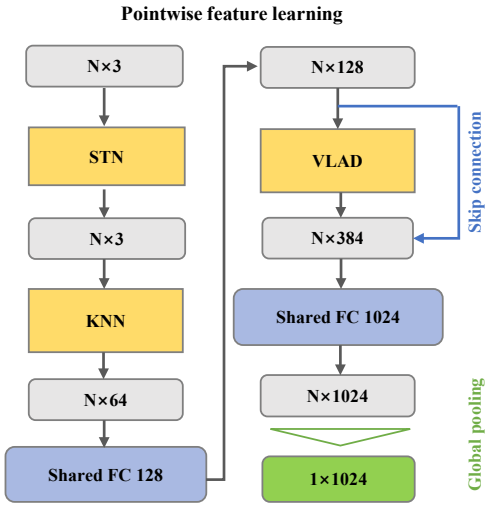


Fig. 4 The pointwise feature learning involves three stages: STN applies transformations like rotation and translation. Then, KNN extracts low-level geometric details for each point in the 3D point cloud.

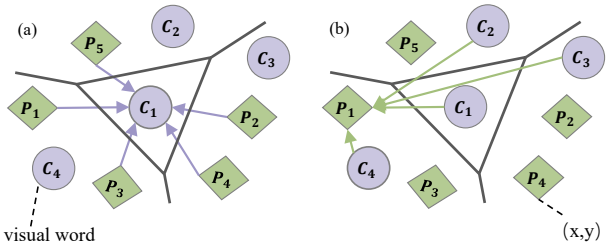


Fig. 5 PointWiseNet's VLAD module indirectly represents high-level semantic features by linking each point's low-level geometric descriptor with select visual words. PointNetVLAD extends PointNet and NetVLAD's success to enable large-scale place recognition using 3D point cloud data retrieval.

uncertainty or disorder of the distribution.

$$z = F_\delta(w(t_0)) \triangleq w(t_0) + \int_{t_0}^{t_1} f_\delta(w(t), t)\, dt, \quad w(t_0) \sim P(w) \tag{10}$$

The flow's continuous-time dynamics (F_δ) are governed by the function f_δ. Earlier, we discussed obtaining the inverse of F_δ through the expression $f_\delta(w(t), t)dt$, where $w(t_1) = z$. The logarithm of the prior distribution's probability can be calculated using this equation:

$$\log P_\delta(z) = \log P\left(F_\delta^{-1}(z)\right) - \int_{t_0}^{t_1} \mathrm{Tr}\left(\frac{\partial f_\delta}{\partial w(t)}\right) dt \tag{11}$$

3.4 Network Structure

The analysis of 3D point cloud structures is notably impacted by both the geometric and semantic details they contain. Nevertheless, employing deep learning for semantic understanding of 3D point clouds presents challenges owing to the inherent unordered nature of the data. Referring to the previous work [42], we use PointWiseNet as the core neural network for fitting point cloud distributions in our diffusion model. The pointwise feature learning process (see Fig. 4) comprises three phases: STN, KNN, and VLAD. Firstly, the STN module applies transformations such as rotation and translation. Secondly, the KNN module extracts the low-level geometric information for each point in the 3D point cloud.

Drawing inspiration from the PointNetVLAD [43] and NetVLAD [44] approaches, PointWiseNet's VLAD module indirectly describes high-level semantic features via the relationship between each point's low-level geometric descriptor and a few visual words. PointNetVLAD has been developed to perform 3D point cloud-based retrieval for large-scale place recognition by building on the success of PointNet and NetVLAD. As shown in Fig. 5 (a), given six 2-dimensional points $p_i|i \in [1, 6]$ as input, and four cluster centers ("visual words" centers) $C_k|k \in [1, 4]$ as VLAD parameters, the output VLAD image representation V is a 4×2 matrix. The (j, k) element of V is computed as follows:

$$V(j,k) = \sum_{i=1}^{6} a_k(p_i)(p_i(d) - c_k(d)) \quad d \in [1, 2] \tag{12}$$

The VLAD module's formal computation involves $p_i(d)$ and $c_k(d)$, representing the d-th dimension of the i-th point and k-th cluster center, respectively. The membership $a_k(p_i)$ signifies the descriptor p_i's association with the visual word c_k. When c_k is the closest cluster to p_i, $a_k(p_i)$ equals 1; otherwise, it's 0. This design in PointNetVLAD aggregates local features into visual words, facilitating the VLAD feature for image retrieval.

For illustration, consider six 2D points $\{p_i|i \in [1, 6]\}$ as VLAD module input (Fig. 5 (b)), with four learnable visual words ($C_k|k \in [1, 4]$) initialized via backpropagation. Each p_i aligns with a visual word C_k, represented by the residual vector $p_i - C_k$, signifying the difference between them. The relationship of the i-th point p_i to the four visual words is denoted as r, and its (i, d) element is computed as follows:

$$\boldsymbol{r}(i,d) = \sum_{k=1}^{4} a_i(c_k)(p_i(d) - c_k(d)) \quad d \in [1, 2] \tag{13}$$

Algorithm 1: Training Algorithm

Data: $q(X^{(0)})$: Point cloud dataset
repeat
  $z = z \in q_\phi(z|X^{(0)})$
  $t = t \in U[1, T]$
  $X^{(0)} = X^{(0)} \in q(X^{(0)})$
  $X^{(t)} = X^{(t)} \in q(X^{(1,T)}|X^{(0)})$
  $L_X := D_{\mathrm{KL}}(q(X_i^{(t-1)}|X_i^{(t)}, X_i^{(0)})\|p_\theta(X_i^{(t-1)}|X_i^{(t)}, z)))$
  $L_z := D_{\mathrm{KL}}(q_\varphi(z|X^{(0)})\|p(z))$
  $\nabla := \nabla_\theta(L_X + L_z)$
  execute gradient descrent
until *Converged*;

3.5 Training objective

Drawing upon the principles of the aforementioned diffusion model, we crafted the training process outlined in Algorithm 1. Subsequently, we will elaborate on our training objectives and loss functions in detail.

The objective of training the reverse diffusion process is to maximize the log-likelihood of the point cloud, denoted as $E[logp_\theta(X^{(0)})]$. However, directly optimizing the exact log-likelihood is intractable. Hence, we maximize a varia-

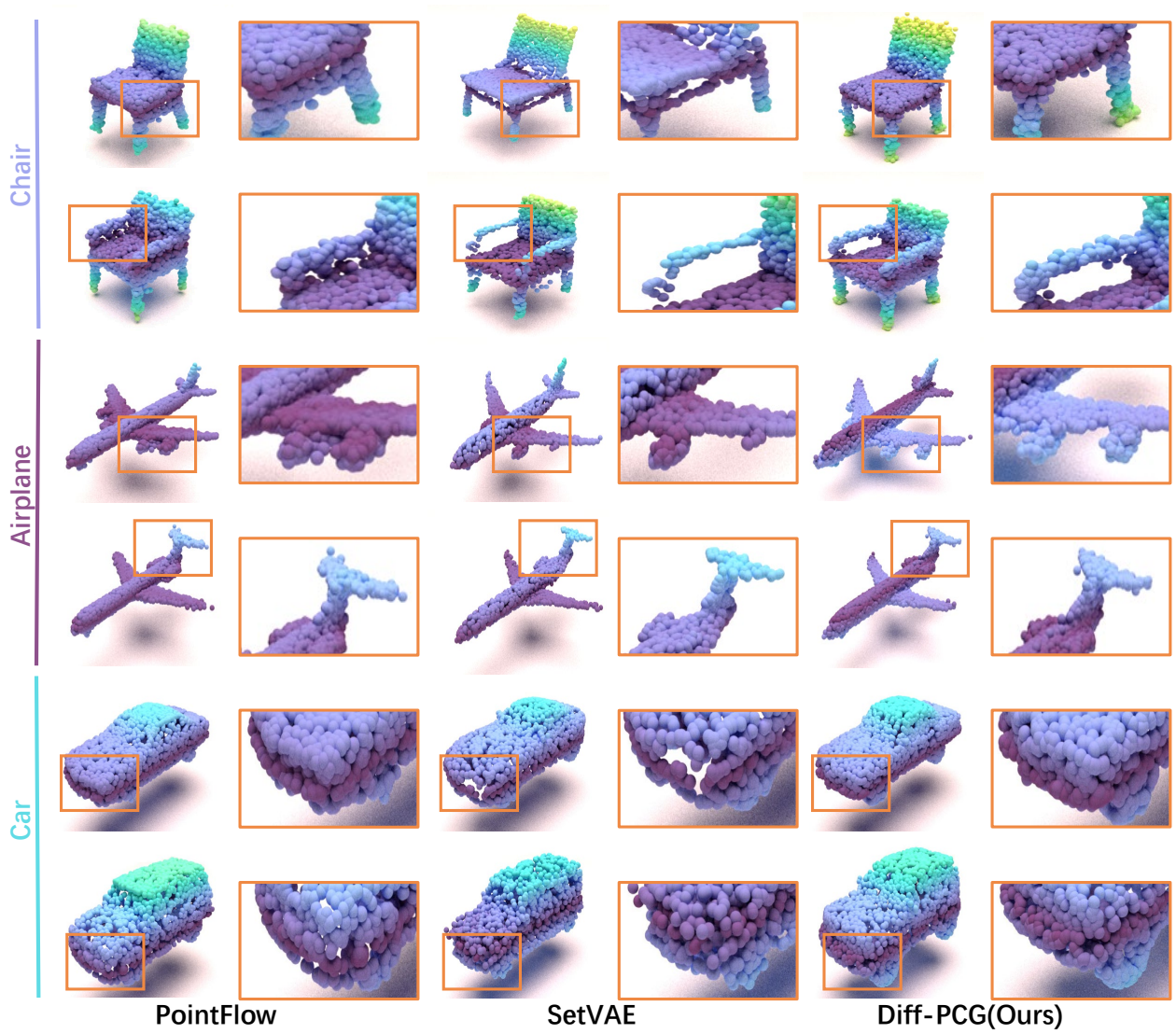


Fig. 6 Generated point clouds from our model and comparative methods are depicted in the examples. The figure illustrates the superiority of our method in accurately representing shape details, like chair armrests, airplane wings, and car tires, compared to previous methods. Each point cloud, rendered by Mitsuba, contains 4096 points.

tional lower bound as a substitute:

$$\begin{aligned}\mathbb{E}\big[\log p_\theta(X^{(0)})\big] &\geq \mathbb{E}_q\Big[\log \frac{p_\theta(X^{(0:T)}, z)}{q(X^{(1:T)}, z|X^{(0)})}\Big] \\ &= \mathbb{E}_q\Big[\log p(X^{(T)}) + \sum_{t=1}^{T} \log \frac{p_\theta(X^{(t-1)}|X^{(t)}, z)}{q(X^{(t)}|X^{(t-1)})} \\ &- \log \frac{q_\varphi(z|X^{(0)})}{p(z)}\Big].\end{aligned} \tag{14}$$

As the mapping is one-to-one and onto, the precise probability of the target distribution can be calculated using the change-of-variable formula:

$$p(z) = p_w(w) \cdot \left|\det \frac{\partial F_\delta}{\partial w}\right|^{-1} \quad \text{where } w = F_\delta^{-1}(z) \tag{15}$$

To enhance the applicability of the aforementioned inequality within our stochastic gradient descent algorithm, we establish it as the training objective, denoted as L, that requires minimization:

$$\begin{aligned}&L(X;\theta,\varphi,\delta) = \mathbb{E}_q[l_p(X;\theta) + l_s(X;\varphi,\delta)], \\ &\mathrm{l}_p(X;\theta) = \textstyle\sum_{t=2}^{T} D_{\mathrm{KL}}\big(q(X^{(t-1)}|X^{(t)}, X^{(0)})\|p_\theta(X^{(t-1)}|X^{(t)}, z)\big), \\ &l_s(X;\varphi,\delta) = D_{\mathrm{KL}}\big(q_\varphi(z|X^{(0)})\|p_w(w) \cdot \left|\det \frac{\partial F_\delta}{\partial w}\right|^{-1}\big).\end{aligned} \tag{16}$$

We can understand this objective as the combination of two components.

Point cloud distribution loss $l_p(X;\theta)$. During the reverse diffusion process, there are two ways to sample the point

Fig. 7 Ablation study was conducted to compare the generation effects of our method under various shape priors. The findings highlight that our approach, utilizing the continuous normalizing flow module as a shape prior learner, surpasses the practice of directly sampling random priors from a Gaussian distribution in preserving intricate shape details.

cloud data at timestep $(t-1)$ from the Gaussian distribution of the point cloud. One method involves precomputing the Gaussian distribution of the point cloud at time $(t-1)$ based on the parameters of the forward diffusion and the point clouds at time 0 and t (denoted as $X^{(0)}$ and $X^{(t)}$). The other method entails using the point cloud at time t and the shape latent variable z as input parameters to train our neural network, which fits the Gaussian distribution of the point cloud at timestep $(t-1)$. The calculation of the Kullback-Leibler (KL) divergence between these two Gaussian distributions forms the point cloud denoising fitting loss.

Shape prior loss $l_s(X;\varphi,\delta)$. The shape prior loss encourages the shape prior learning module, which is our continuous normalization flow module as discussed earlier, to learn the complex Gaussian distribution of the point cloud shape prior. This enables the transformation of randomly sampled latent variables from a standard Gaussian distribution into shape latent variables conforming to a certain shape's complex Gaussian distribution in later stages.

4 Experiments

4.1 Experimental Setup

Dataset. ShapeNet [45] is a large-scale 3D shape benchmark that provides a comprehensive collection of 3D models across different object categories. According to the official website of ShapeNet, the dataset covers several main categories of 3D models. ShapeNetCore: This is the core part

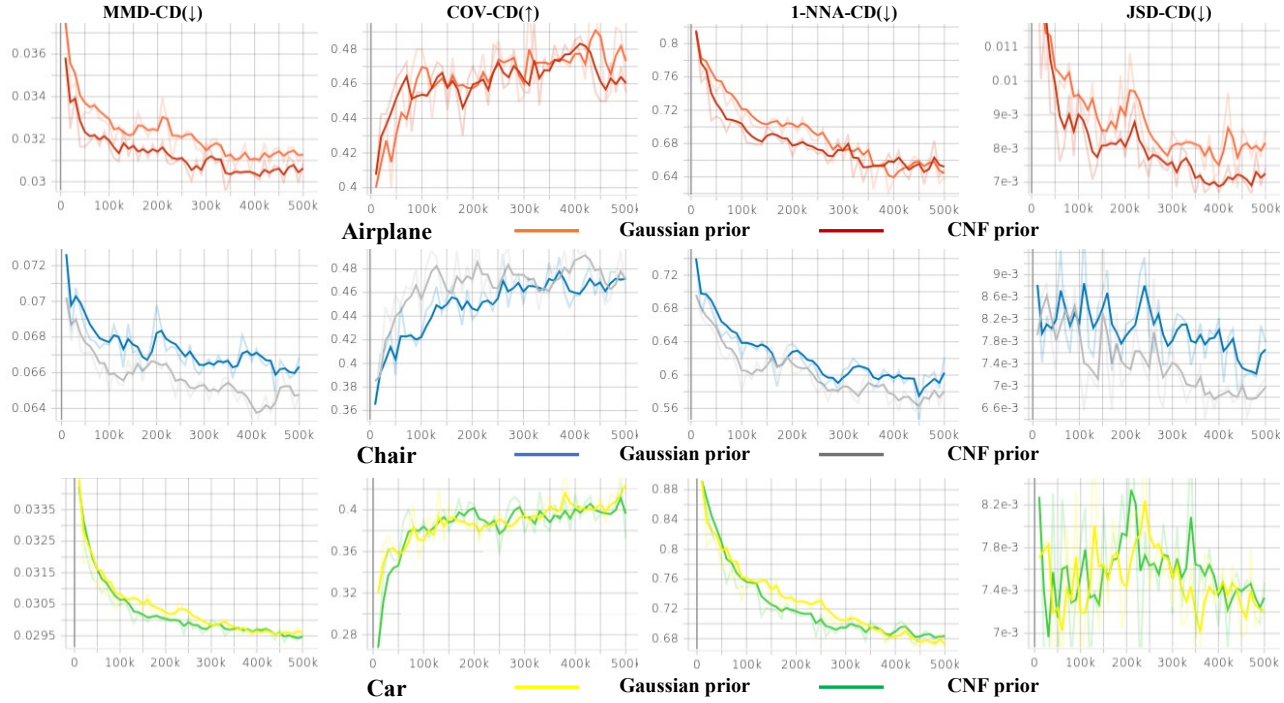


Fig. 8 Empirical evaluation of diverse shape generation scenarios employing distinct prior assessment metrics, with variations in the number of iterations. (For visual considerations, we accept Tensorboard's smoothing of curves by default, and the original data curves are displayed in the background of the chart with a certain transparency.)

of the ShapeNet dataset, containing around 55 object categories, covering various common object categories such as furniture, vehicles, and electronics. In our point cloud generation task, we employ the ShapeNetCore dataset to train our model. Referring to mainstream practices in this field, we have selected three types of point clouds: Airplane, chair, and car for more detailed quantitative and visual evaluation.

Evaluation Metrics. Several metrics have been proposed in the literature to assess the quality of generated point clouds quantitatively. However, some of these metrics have inherent limitations. Building upon the work of Yang et al. [24], it has been observed that metrics such as COV (Coverage) and MMD (Maximum Mean Discrepancy) may not provide reliable evaluations of point cloud generation performance. In contrast, the 1-Nearest Neighbor Accuracy (1-NNA) metric has been identified as a more suitable measure. Additionally, the Jensen-Shannon Divergence (JSD) is occasionally employed as an alternative metric for evaluating point cloud generation performance. Following the recommendations of Yang et al. [24] and Zhou et al. [23], we adopt 1-NNA as our primary evaluation metric to assess the quality of point cloud generation. Given a generated set of point clouds S_g and a reference set S_r, the detailed formula for these metrics is as follows :

$$\text{MMD}(S_g, S_r) = \frac{1}{|S_r|} \sum_{Y \in S_r} \min_{X \in S_g} D(X, Y) \tag{17}$$

where D(·, ·) is either the Chamfer distance (CD) or earth mover distance (EMD).COV is a metric that quantifies the number of reference point clouds that are successfully matched to at least one generated shape.

$$\text{COV}(S_g, S_r) = \frac{|\{\arg\min_{Y \in S_r} D(X, Y) | X \in S_g\}|}{|S_r|} \tag{18}$$

where D(·, ·) is again either CD or EMD. The concept behind MMD is to compute the average distance between the point clouds in the reference set and their nearest neighbors in the generated set. In order to address the limitations of COV and MMD, Yang et al. [10] introduced the use of 1-NNA (one nearest neighbor accuracy) as a metric for evaluating point cloud generative models:

$$\text{1-NNA}(S_g, S_r) = \frac{\sum_{X \in S_g} \mathbb{I}[N_X \in S_g] + \sum_{Y \in S_r} \mathbb{I}[N_Y \in S_r]}{|S_g| + |S_r|} \tag{19}$$

where I[·] is the indicator function and N_X is the nearest neighbor of X in the set $S_r \cup S_g - X$. Moreover, we employ both Chamfer Distance (CD) and Earth Mover's Distance (EMD) to compute these metrics, following common definitions in the field of point cloud generation:

$$\text{CD}(X, Y) = \sum_{x \in X} \min_{y \in Y} ||x - y||_2^2 + \sum_{y \in Y} \min_{x \in X} ||x - y||_2^2 \tag{20}$$

$$\mathrm{EMD}(X,Y) = \min_{\gamma:X\to Y} \sum_{x\in X} ||x-\gamma(x)||_2 \tag{21}$$

It is essential to acknowledge that within the domain of deep learning, the fluctuation in the number of point clouds holds considerable importance. Despite this variability, researchers often prioritize the geometric quality of the point cloud rather than focusing solely on quantity. To ensure enhanced comparability and simplify the issue at hand, the customary approach involves employing the Chamfer Distance (CD) formula cumulatively, without normalizing based on the quantity of point clouds.

4.2 Comparison

In our pursuit of advancing point cloud generation, we embarked on a rigorous comparative analysis pitting our model against several notable prior works: TreeGAN [46], SP-GAN [47], SetVAE [48], and PointFlow [10]. We conducted comparative assessments within the point cloud generation task, employing three specific shapes (cars, airplanes, and chairs) extracted from the ShapeNet dataset [45] as our benchmark. While recognizing the capability of our method to generate objects across a wider range within the ShapeNet dataset, we chose to concentrate on these three representative shapes for the sake of brevity and to avoid redundancy.

The results of these comparative experiments, which involved the incorporation of continuous normalizing flow as a prior learning component, suggest that our diffusion-based point cloud generation framework exhibits competitive performance compared to traditional VAE-based generation methods and models relying solely on continuous normalizing flow as a generation module.

These observations are visually represented in Fig. 6, offering a compelling demonstration of our model's proficiency in point cloud generation. Our approach consistently demonstrates effectiveness, not only in generating point clouds but also in preserving intricate shape details. These details include elements such as airplane wings, chair armrests, and car wheels, as highlighted in Fig. 6.

Quantitative assessments provide additional support for the effectiveness of our model. We have diligently tabulated these results in Tab. 1, enhancing clarity by scaling the values of the Chamfer Distance (CD) and Earth Mover's Distance (EMD) indices for easy interpretation. Our method demonstrates competitive performance across these metrics, indicating its ability to generate point clouds that can be on par with, and in some cases, surpass those generated by earlier methods. The quantitative evaluation results are affected by factors such as data shape, training parameters, and training conditions, and vary greatly in different situations. As

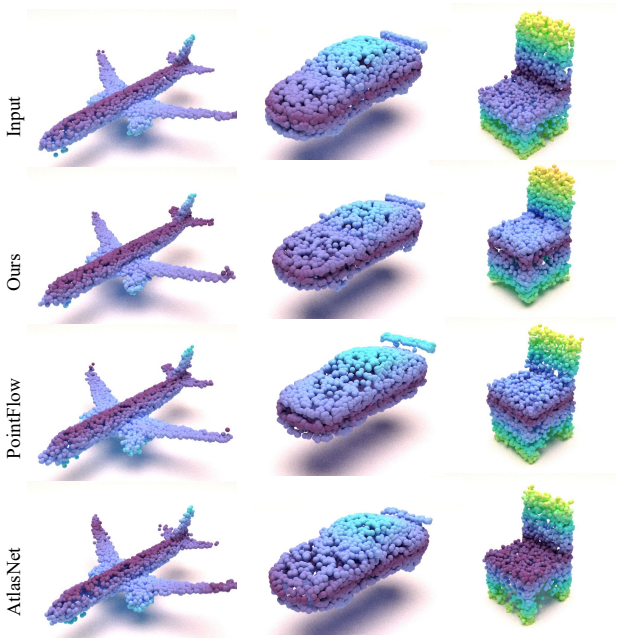


Fig. 9 Reconstructed point clouds from various auto-encoders illustrate the robust performance of our autoencoder module in reconstructing point clouds, providing additional confirmation regarding the efficacy of our encoder and shape prior learning modules.

can be seen from the table, our method achieved good results for the category of Airplanes and Chairs. Although it has not always achieved the best results for the category of Car, the visualization results show that the point cloud generated by our method still retains good shape details.

4.3 Ablation study

To validate the effectiveness of the autoencoder module and shape prior learning module utilized in our approach, we designed the following ablation experiments.

Shape prior learning: In our efforts to advance point cloud generation, we turned to Yang et al. [10] on continuous normalizing flow (CNF) as a vital component for learning the latent variables that govern the shape prior of point clouds, as referenced in their study. To emphasize the importance of this module, we conducted a series of experiments to assess its significance. In these experiments, we substituted CNF with a fixed Gaussian prior, allowing for direct comparisons with conventional generative approaches.

The results of these experiments are visually represented in Fig. 7, providing visual comparisons that highlight the substantial impact of our approach. We further quantified our findings, as presented in Tab. 2, using various quantitative metrics. Notably, we explain the transformation process of the Chamfer Distance (CD) metric for both shape priors during the training process in Fig. 8.

Table 1 Quantitative evaluation of multiple models for generating point clouds. MMD-CD is multiplied with 1×10^3, and MMD-EMD is multiplied with 1×10^2.

Shape	Model	MMD(↓)		COV(%↑)		1-NNA(%↓)		JSD(↓)
		CD	EMD	CD	EMD	CD	EMD	
Airplane	TreeGAN [46]	0.558	1.460	31.85	17.78	97.53	99.88	15.646
	SP-GAN [47]	0.403	0.766	26.42	24.44	94.69	93.95	9.472
	SetVAE [48]	**0.200**	0.367	43.70	**48.40**	76.54	**67.65**	2.369
	PointFlow [10]	0.224	0.390	47.90	46.41	75.68	70.74	**1.536**
	Diff-PCG(Ours)	0.294	**0.303**	**49.50**	47.72	**65.12**	75.45	6.580
Chair	TreeGAN [46]	4.841	3.504	39.88	26.59	88.37	96.37	13.282
	SP-GAN [47]	4.208	2.620	40.03	32.93	72.58	83.69	10.769
	SetVAE [48]	**2.545**	7.820	46.98	45.01	58.76	**61.48**	8.242
	PointFlow [10]	13.631	1.856	41.86	43.38	66.13	68.40	12.470
	Diff-PCG(Ours)	6.311	**1.341**	**50.75**	**48.32**	**55.62**	62.31	**6.360**
Car	TreeGAN [46]	1.142	1.063	40.06	31.53	89.77	94.89	5.850
	SP-GAN [47]	1.168	1.021	34.94	31.82	87.36	85.94	7.321
	SetVAE [48]	**0.882**	**0.733**	**48.58**	44.60	59.66	63.35	1.322
	PointFlow [10]	0.901	0.807	46.88	**50.00**	**58.10**	**56.25**	0.870
	Diff-PCG(Ours)	2.502	2.115	43.75	40.02	66.50	69.27	**0.781**

The experimental results illustrate a compelling conclusion: our method, based on the utilization of continuous normalized flow as a shape prior learner, outperforms the conventional practice of directly sampling shape priors from a Gaussian distribution. This superiority is particularly evident in intricate details such as chair armrests and legs, airplane wings and tails, and car bodies, as demonstrated in Fig. 7. Additionally, our approach consistently delivers superior results across a range of quantitative metrics, emphasizing its ability to generate point cloud shapes that are not only more accurate but also more stable.

Table 2 Ablation analysis. Our method combines different shapes prior in generative experiments.

Model	MMD(↓)		COV(%, ↑)		1-NNA(%, ↓)	
	CD	EMD	CD	EMD	CD	EMD
Airplane(Gaussian)	0.304	1.265	**51.00**	**49.83**	**62.12**	77.31
Airplane(**CNF**)	**0.294**	**0.303**	49.5	47.72	65.12	**75.45**
Chair(Gaussian)	6.472	1.462	49.25	47.81	**54.37**	63.44
Chair(**CNF**)	**6.311**	**1.341**	**50.75**	**48.32**	55.62	**62.31**
Car(Gaussian)	2.999	2.728	41.50	39.21	71.63	83.66
Car(**CNF**)	**2.502**	**2.115**	**43.75**	**40.02**	**66.50**	**69.24**

Table 3 Comparison of auto-encoding performance of point clouds between Atlas(S1) representing the l-sphere variant and Atlas(P25) representing the 2.5-square variant of AtlasNet. CD is scaled by a factor of 1000, while EMD is scaled by a factor of 100.

Dataset	Metric(↓)	Atlas(S1) [49]	Atlas(P25) [49]	PointFlow [10]	Ours
ShapeNet	CD	5.873	**5.420**	7.550	6.574
	EMD	5.457	5.599	5.172	**5.112**
Chair	CD	5.479	**4.980**	6.795	5.632
	EMD	5.550	5.282	**5.008**	5.447
Car	CD	6.906	6.503	5.828	**5.637**
	EMD	5.617	5.408	**4.390**	5.145
Airplane	CD	2.000	**1.795**	2.420	2.112
	EMD	4.311	4.366	**3.311**	**3.433**

Table 4 Unsupervised feature learning begins with training models on ShapeNet to grasp shape representations, subsequently assessing their performance on ModelNet40 and ModelNet10.

Method	ModelNet10(↑ %)	ModelNet40(↑ %)
LFD [50]	75.5	79.9
SPH [51]	68.2	79.8
VConv-DAE [52]	75.5	80.5
FoldingNet [53]	**88.4**	94.4
3D-GAN [2]	83.3	91.0
1-GAN [5]	84.5	**95.4**
PointFlow [10]	86.8	93.7
Ours	**87.1**	92.8

Auto-Encoder Learning: We evaluate the efficacy of the proposed auto-encoder in reconstructing point clouds by conducting a comparative analysis with relevant models such as AtlasNet [49] and PointFlow [10]. Our comprehensive assessment covers four datasets, encompassing three ShapeNet categories—airplane, car, chair—and the entirety of the ShapeNet dataset.

Our method's performance, as outlined in Tab. 3, demonstrates competitive outcomes compared to existing approaches when evaluated using metrics such as Earth Mover's Distance (EMD). It can be observed that our approach achieves competitive metric results in most scenarios, outperforming similar methods in the CD metric for the Car category and in the EMD metric for the entire ShapeNet dataset. Furthermore, the visualization of reconstructed point clouds in Fig. 9 confirms our model's effectiveness in the reconstruction task.

We proceed by assessing the capacity of our auto-encoders in learning representations. Specifically, we derive the latent representations from our auto-encoder trained on

the complete ShapeNet dataset. Subsequently, we employ a linear SVM classifier atop these representations, utilizing either ModelNet10 or ModelNet40 [54]. It's important to note that, uniquely for this task, we standardize each point cloud to achieve zero mean per axis and global unit variance, aligning with established methodologies [55]. Additionally, we introduce random rotations along the gravity axis during the auto-encoder training phase. This evaluation involves comparing the accuracy of off-the-shelf SVMs, which are trained to utilize the acquired representations.

As can be seen from Tab. 4, although our method has a slight gap compared to the current best-performing method, it is undeniable that it remains competitive and at a relatively high level among similar approaches.

5 Conclusion and Future Work

We propose a novel probabilistic framework, namely Diffusion Point Cloud Generation Conditioned on Continuous Normalizing Flow (Diff-PCG), inspired by the Denoising Diffusion Probabilistic Model (DDPM). Our model expands upon the capabilities of DDPM by integrating the PVCNN encoder and incorporating a learnable shape prior rooted in Continuous Normalizing Flow (CNF), thereby enhancing the representational capacity of the latent variable encoding. Extensive experiments validate that our model achieves excellent results in generating high-quality point clouds with remarkable fidelity.

Future work includes: (i) expanding the range of currently generated point cloud shapes to include a greater variety of object shapes; (ii) increasing constraint adjustment, such as image or text-based priors for point cloud generation; (iii) further enhancing control over point cloud details to ensure the stability of generated shapes.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (Nos. U22B2034, U21A20515, 62376271, 62172416, 62262043, 62102414, 62365014, and 62162044), in part by Beijing Natural Science Foundation(No. L231013), and the Open Project Program of State Key Laboratory of Virtual Reality Technology and Systems, Beihang University (No. VRLAB2023B01).

References

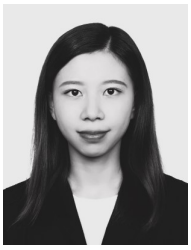
1. Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *Advances In Neural Information Processing Systems*, 35:31841–31854, 2022. 2
2. Jiajun Wu, Chengkai 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. 2, 11
3. Chun-Liang Li, Manzil Zaheer, Yang Zhang, Barnabas Poczos, and Ruslan Salakhutdinov. Point cloud gan. *arXiv preprint arXiv:1810.05795*, 2018. 2, 3
4. Wenlong Huang, Brian Lai, Weijian Xu, and Zhuowen Tu. 3d volumetric modeling with introspective neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8481–8488, 2019. 2
5. 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. 2, 11
6. Aditya Sanghi, Rao Fu, Vivian Liu, Karl Willis, Hooman Shayani, Amir Hosein Khasahmadi, Srinath Sridhar, and Daniel Ritchie. Textcraft: Zero-shot generation of high-fidelity and diverse shapes from text. *arXiv preprint arXiv:2211.01427*, 2022. 2
7. Or Litany, Alex Bronstein, Michael Bronstein, and Ameesh Makadia. Deformable shape completion with graph convolutional autoencoders. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1886–1895, 2018. 2
8. Qingyang Tan, Lin Gao, Yu-Kun Lai, and Shihong Xia. Variational autoencoders for deforming 3d mesh models. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5841–5850, 2018. 2
9. 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. 2
10. 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. 2, 9, 10, 11
11. 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. 2, 3
12. Charlie Nash, Yaroslav Ganin, SM Ali Eslami, and Peter Battaglia. Polygen: An autoregressive generative model of 3d meshes. In *International conference on machine learning*, pages 7220–7229. PMLR, 2020. 2
13. Wei-Jan Kol, Chen-Yi Chiu, Yu-Liang Kuo, and Wei-Chen Chiu. Rpg: learning recursive point cloud generation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 544–551. IEEE, 2022. 2
14. Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy Mitra, and Leonidas J Guibas. Structurenet: Hierarchical graph networks for 3d shape generation. *arXiv preprint arXiv:1908.00575*, 2019. 2
15. Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. 43(12):4338–4364, 2020. 2
16. Zhen Liu, Yao Feng, Michael J Black, Derek Nowrouzezahrai, Liam Paull, and Weiyang Liu. Meshdiffusion: Score-based generative 3d mesh modeling. *arXiv preprint arXiv:2303.08133*, 2023. 2

17. Jiong Yang, Jian Zhang, Zhengyang Cai, and Dongyang Fang. Novel 3d local feature descriptor of point clouds based on spatial voxel homogenization for feature matching. *Visual Computing for Industry, Biomedicine, and Art*, 6(1):18, 2023. 2
18. Yiyi Liao, Katja Schwarz, Lars Mescheder, and Andreas Geiger. Towards unsupervised learning of generative models for 3d controllable image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5871–5880, 2020. 2
19. Zhiqin Chen, Vladimir G Kim, Matthew Fisher, Noam Aigerman, Hao Zhang, and Siddhartha Chaudhuri. Decor-gan: 3d shape detailization by conditional refinement. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15740–15749, 2021. 2
20. Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022. 2
21. Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019. 2
22. 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. 2
23. 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. 2, 3, 9
24. Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. *Advances in Neural Information Processing Systems*, 32, 2019. 2, 3, 9
25. 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. 2, 4
26. Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019. 2
27. Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 2
28. Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 2
29. Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 9621–9630, 2019. 2
30. 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. 2
31. Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 2
32. Pu Li, Jianwei Guo, Xiaopeng Zhang, and Dong-Ming Yan. Secad-net: Self-supervised cad reconstruction by learning sketch-extrude operations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16816–16826, 2023. 2
33. Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2018. 2
34. Shidi Li, Miaomiao Liu, and Christian Walder. Editvae: Unsupervised parts-aware controllable 3d point cloud shape generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1386–1394, 2022. 2
35. Lei Wang, Yuchun Huang, Pengjie Tao, Yaolin Hou, and Yuxuan Liu. Learning geometry-image representation for 3d point cloud generation. *arXiv preprint arXiv:2011.14289*, 2020. 2
36. Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017. 3
37. Zhulun Yang, Yijun Chen, Xianwei Zheng, Yadong Chang, and Xutao Li. Conditional gan for point cloud generation. In *Proceedings of the Asian Conference on Computer Vision*, pages 3189–3205, 2022. 3
38. An-Chieh Cheng, Xueting Li, Sifei Liu, Min Sun, and Ming-Hsuan Yang. Autoregressive 3d shape generation via canonical mapping. In *European Conference on Computer Vision*, pages 89–104. Springer, 2022. 3
39. Zhe Li, Zhangyang Gao, Cheng Tan, Stan Z Li, and Laurence T Yang. General point model with autoencoding and autoregressive. *arXiv preprint arXiv:2310.16861*, 2023. 3
40. Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015. 5
41. Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018. 5
42. Dejun Zhang, Fazhi He, Zhigang Tu, Lu Zou, and Yilin Chen. Pointwise geometric and semantic learning network on 3d point clouds. *Integrated Computer-Aided Engineering*, 27(1):57–75, 2020. 6
43. Mikaela Angelina Uy and Gim Hee Lee. Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4470–4479, 2018. 6
44. Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307, 2016. 6
45. 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. 8, 10
46. 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, 11
47. Ruihui Li, Xianzhi Li, Ka-Hei Hui, and Chi-Wing Fu. Sp-gan: Sphere-guided 3d shape generation and manipulation. *ACM Transactions on Graphics (TOG)*, 40(4):1–12, 2021. 10, 11
48. Jinwoo Kim, Jaehoon Yoo, Juho Lee, and Seunghoon Hong. Setvae: Learning hierarchical composition for generative modeling of set-structured data, 2021. 10, 11
49. Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. Atlasnet: A papier-mâché approach to learning 3d surface generation, 2018. 11
50. Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, volume 22, pages 223–232. Wiley Online Library, 2003. 11
51. Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3 d shape descriptors. In *Symposium on geometry processing*, volume 6, pages 156–164, 2003. 11
52. Abhishek Sharma, Oliver Grau, and Mario Fritz. Vconv-dae: Deep volumetric shape learning without object labels. In *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14*, pages 236–250. Springer, 2016. 11

53. Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 206–215, 2018. 11
54. Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, 2015. 12
55. Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 12

Ting Yu received the B.S. degree in Mechanical Engineering from Beijing Jiaotong University, China, in June 2021. He is currently an M.S. student majoring in Computer Application Technology in State Key Laboratory of Multimodal Artificial Intelligence Systems at Institute of Automation, Chinese Academy of Sciences. His research interests include computer vision and point cloud processing.

Weiliang Meng received his Ph.D. degree in Computer Application from Institute of Software, Chinese Academy of Sciences in 2010. He is currently an Associate Professor in State Key Laboratory of Multimodal Artificial Intelligence Systems at Institute of Automation, Chinese Academy of Sciences. His main research fields include artificial intelligence, computer vision, 3D scene analysis, 3D geometry processing, computer graphics.



Zhongqi Wu received her Ph.D. degree from National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences in 2022. She is currently a Postdoctoral researcher in Institute of Science and Development, Chinese Academy of Sciences. Her research interests include image processing and computer vision.

Jianwei Guo is an Associate Professor at the State Key Laboratory of Multimodal Artificial Intelligence Systems (MAIS), Institute of Automation, Chinese Academy of Sciences (CASIA). He received his Ph.D. degree in computer science from CASIA in 2016, and his Bachelor's degree from Shandong University in 2011. His research interests include computer graphics, 3D vision, and geometry processing.

Xiaopeng Zhang received the B.S. degree and M.S. degree in Mathematics from Northwest University in 1984 and 1987 respectively, and the Ph.D. degree in Computer Science from Institute of Software, Chinese Academy of Sciences in 1999. He is currently a Professor with the State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences. His main research interests are computer graphics and computer vision.