StrucADT: Generating Structure-controlled 3D Point Clouds with Adjacency Diffusion Transformer

Zhenyu Shu, Jiajun Shen*, Zhongui Chen, Xiaoguang Han, and Shiqing Xin

Abstract—In the field of 3D point cloud generation, numerous 3D generative models have demonstrated the ability to generate diverse and realistic 3D shapes. However, the majority of these approaches struggle to generate controllable 3D point cloud shapes that meet user-specific requirements, hindering the large-scale application of 3D point cloud generation. To address the challenge of lacking control in 3D point cloud generation, we are the first to propose controlling the generation of point clouds by shape structures that comprise part existences and part adjacency relationships. We manually annotate the adjacency relationships between the segmented parts of point cloud shapes, thereby constructing a StructureGraph representation. Based on this StructureGraph representation, we introduce StrucADT, a novel structure-controllable point cloud generation model, which consists of StructureGraphNet module to extract structure-aware latent features, cCNF Prior module to learn the distribution of the latent features controlled by the part adjacency, and Diffusion Transformer module conditioned on the latent features and part adjacency to generate structure-consistent point cloud shapes. Experimental results demonstrate that our structure-controllable 3D point cloud generation method produces high-quality and diverse point cloud shapes, enabling the generation of controllable point clouds based on user-specified shape structures and achieving state-of-the-art performance in controllable point cloud generation on the ShapeNet dataset.

Index Terms—3D point cloud generation, Structure control, Diffusion Transformer

1 Introduction

3D shape generation [1], [2] is a fundamental task in computer graphics, holding significant importance across various domains, including modeling, animation, and gaming industries. The field of 3D shape generation has witnessed substantial progress, with numerous state-of-the-art methods and applications emerging in voxels [3], [4], [5], [6], [7], [8], point clouds [9], [10], [11], [12], meshes [13], [14], [15], [16], [17], [18], and structured representations [19], [20], [21], [22], [23]. 3D point cloud shape data can be easily obtained through 3D scanners, making the generation of 3D point cloud shapes a topic of extensive attention and research.

In the realm of 3D point cloud generation, many 3D generative models, such as PointFlow [9], DPM [10], Point-E [11], and DiffFacto [12], have shown proficiency in producing diverse and realistic 3D shapes. Methods like PointFlow, DPM, and DiffFacto treat the 3D point cloud generation process as a distribution of distributions [9], first sampling from the overall shape distribution of a class of point clouds and then sampling the distribution of points from the given shape, thus enabling the generation of point clouds with an arbitrary number of points. These methods employ flow-based models [24], [25], [26], [27] or diffusion models [28], [29], [30], [31], [32], [33], [34], [35] to learn the aforementioned distributions.

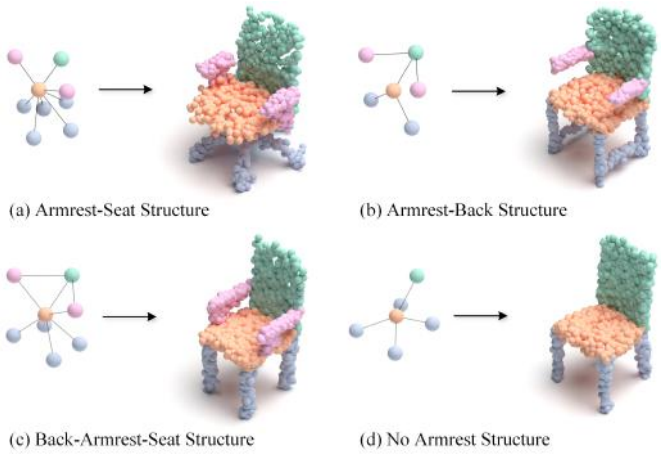

Fig. 1. (a) The Armrest-Seat structure to control the chair generated with the armrest attached to the seat, (b) the Armrest-Back structure to control the chair generated with the armrest attached to the back, (c) the Back-Armrest-Seat structure to control the chair generated with the armrest attached to the back and the seat, and (d) the No-Armrest structure to control the chair generated without the armrest.

However, most of these approaches struggle to generate controllable 3D point cloud shapes. These methods can only learn to approximate the shape distribution from the

- *Zhenyu Shu is with School of Computer and Data Engineering, NingboTech University, Ningbo 315100, China. He is also with Ningbo Institute, Zhejiang University, Ningbo 315100, China (e-mail: shuzhenyu@nit.zju.edu.cn).*
- *Jiajun Shen is with School of Software Technology, Zhejiang University, Ningbo 315048, China (e-mail: jiajunshen_paper@163.com).*
- *Zhonggui Chen is with the School of Informatics, Xiamen University, Xiamen 361005, China (e-mail: chenzhonggui@xmu.edu.cn).*
- *Xiaoguang Han is with Shenzhen Research Institute of Big Data, Chinese University of Hong Kong, Shenzhen 518172, China. (e-mail: hanxiaoguang@cuhk.edu.cn).*
- *Shiqing Xin is with School of Computer Science and Technology, ShanDong University, Jinan 250100, China (e-mail: xinshiqing@sdu.edu.cn).*

Manuscript received month day, year; revised month day, year.

dataset and sample point cloud shapes from the learned distribution, but they lack the ability to control the generation of shapes that meet user-specific requirements, as users usually want to generate shapes that fit their needs. Some multimodal generation methods, such as Point-E [11], can control the generation process with text or image conditions. However, these methods require text or image data matching the point cloud shapes, thus necessitating extensive manual annotation, which is costly and challenging.

In order to address the challenge of lacking control in 3D point cloud generation, we explore the inherent structure of 3D shapes. The structure of a shape is whether the parts that make up the shape appear and whether these parts are adjacent. For shapes within the same category, although their overall geometric profile may vary significantly, their structures remain similar [36]. For example, a chair is typically composed of the back, seat, legs, and armrests, where the seat is usually adjacent to the back and legs. The core observation of this paper is leveraging the structure of 3D point cloud shapes to better control their generation. Figure 1 shows that different structures of the chair are able to control the generation of chairs with different styles: a chair generated with armrests attached to the seat or the back or both the seat and the back, or even without the armrests. By introducing the shape structure, it becomes possible to control the generation of 3D point cloud shapes explicitly and more precisely.

However, existing 3D shape segmentation datasets [22], [37], [38], [39], [40] only provide semantic segmentation labels without the connectivity information between parts. Although StructureNet [20] represents shapes as n-ary part hierarchies with adjacency between sibling parts, they simply compute the minimum distance between two segmented parts to obtain the part adjacency on the PartNet [22] dataset. In contrast, we manually annotate the adjacency relationships between segmented parts on the widely used ShapeNet dataset, forming *StructureGraph* representation for point cloud shapes to control their generation more precisely. In this graph, nodes represent the segmented parts of a point cloud shape, while edges denote the connections between these segmented parts.

To control the 3D point cloud generation process with our *StructureGraph* representation, we introduce **StrucADT**: Generating **Struc**ture-controlled 3D point clouds with **A**djacency **D**iffusion **T**ransformer. StrucADT explicitly uses the StructureGraph of 3D shapes as user-specific input to control the 3D point cloud generation. To enable the network to incorporate the feature information from the StructureGraph, we design a *StructureGraphNet* encoder to fuse the structural information between parts. The learned features of the StructureGraph and the part adjacency are then incorporated into our proposed *cCNF Prior* module and *Diffusion Transformer* [41], [42] module to generate structure-consistent point cloud shapes.

Our contributions are as follows:

- We introduce StrucADT: Generating Structure-controlled 3D point clouds with Adjacency Diffusion Transformer, which can produce novel and realistic point cloud shapes controlled by user-specific shape structures. To the best of our knowledge, we are the first to control the 3D point cloud generation with shape structures.
- To better control the generation of 3D point clouds using the structure of 3D point cloud shapes, we annotate the adjacency relationships between segmented parts on the ShapeNet dataset, forming *StructureGraph* representation for 3D shapes.
- We propose *StructureGraphNet* module to extract structure-aware latent features, *cCNF Prior* module to learn the distribution of the latent features controlled by the part adjacency, and *Diffusion Transformer* module conditioned on the latent features and part adjacency to generate structure-controllable point cloud shapes. Extensive experimental results on the ShapeNet dataset showcase that our method achieves state-of-the-art performance in controllable point cloud generation.

The rest of the paper is structured as follows. Section 2 provides a review of the related work. Section 3 presents a detailed explanation of our proposed method. In Section 4, we evaluate the performance of our algorithm on the ShapeNet dataset. Section 5 discusses the limitations of our approach and suggests future research directions. Lastly, Section 6 concludes the paper.

2 Related Work

2.1 3D Point Cloud Generation

Achlioptas et al. [43] propose a deep autoencoder network with good reconstruction quality and generalization ability. The network includes an r-GAN running on the original point cloud, an l-GAN with significant improvements trained in the fixed latent space of the autoencoder, and the Gaussian Mixture Models (GMMs). Valsesia et al. [44] improve the r-GAN using a generative adversarial network architecture based on graph convolutional networks. This method can dynamically construct a proximity graph during the generation process, enabling the extraction and representation of local features. TreeGAN [45] enhances feature representation capabilities by performing graph convolutions in a tree, utilizing ancestor information to generate multi-category 3D point clouds in an unsupervised manner.

Most of the above-mentioned 3D point cloud generation models can only generate a fixed number of point clouds. To address this issue, PointFlow [9] views the 3D point cloud generation process as sampling a shape from a shape distribution and then sampling a point cloud from the distribution of that shape. The method uses two CNF [27] generation models to model the above process, enabling the generation of point clouds with an arbitrary number of points. Based on PointFlow, DPM [10] proposes a 3D point cloud generation method built up on diffusion models that model the reverse diffusion process of point clouds as a Markov chain conditioned on shape latent variables.

However, most of these approaches struggle to generate controllable 3D point cloud shapes. To address the challenge of lacking control in 3D point cloud shape generation, we introduce StructureGraph representation and propose a novel part adjacency conditioned Diffusion Transformer module to control the process of 3D point cloud generation.

2.2 Structure-Aware 3D Shape Generation

The structure of a shape generally refers to the components that constitute a shape and the connections between these components, which is a high-level abstraction of 3D shapes. In order to enable users to modify or manipulate the synthesized 3D shapes easily, the generative models should be structure-aware, and users should be able to manipulate the generated 3D shapes by modifying the high-level structure of the shapes [36].

Many works represent the structure of shapes as trees or graphs. GRASS [19] proposes a new neural network architecture for encoding and synthesizing 3D shapes, particularly their structures. The method employs a recursive neural networks (RvNNs) based autoencoder to map flat, unlabeled, arbitrary parts to compact codes. The code effectively captures the hierarchy of man-made 3D objects with varying structural complexity. Ren et al. [46] introduce a method for creating consistent visualizations of collections of segmented meshes by embedding graphs jointly, using spectral graph drawing for initialization and stress majorization for refinement, enabling distance preservation and alignment even with partial or soft node correspondences. StructureNet [20] is a hierarchical graph network designed to encode and generate diverse, realistic 3D shapes by representing them as n-ary graphs, effectively handling continuous deformations and structural alterations. SDM-Net [21] proposes a deep generative neural network for producing structured deformable meshes. The architecture of SDM-Net is a two-level variational autoencoder, ensuring consistency between the overall shape structure and surface details. DSG-Net [23] employs a deep neural network that enhances 3D shape generation by learning a disentangled structured and geometric mesh representation, allowing for sophisticated control over shape synthesis with separate yet synergistic encoding of geometry and structure.

Recent works also focus on 3D shape generation and manipulation using implicit representations, emphasizing part-level approaches to enable precise and flexible editing of shapes. SPAGHETTI [47] develops a novel generative framework designed for editing and manipulating 3D shapes represented as neural implicit fields. The architecture disentangles shape representations into intrinsic and extrinsic components, facilitating precise part-level control. SALAD [48] proposes a novel 3D generative model designed for high-quality 3D shape generation and manipulation. It employs a cascaded diffusion framework based on part-level implicit representations, enabling both realistic shape generation and intuitive part-level editing without additional training. 3DShape2VecSet [49] introduces a novel neural field-based representation for 3D shape generation, optimized for generative diffusion models. It encodes 3D shapes into a compact latent space using a set of latent vectors and processes them with cross-attention and self-attention mechanisms.

Although most of these structure-aware methods can generate 3D shapes with plausible structures that are easy to manipulate and modify, their generation process can not be controlled by user-specific input. In contrast, our method can generate realistic and diverse point clouds controlled by the input shape structures.

2.3 Controllable 3D Shape Generation

Many recent works control the generation of 3D shapes by introducing additional conditions such as text and images.

PSGN [50] addresses the issue of non-uniqueness of the true 3D point cloud corresponding to a single image by proposing a novel conditional shape sampler. This sampler is capable of predicting multiple plausible 3D point clouds from an input image. Point-E [11] proposes two diffusion models to generate point clouds from complex prompts. It first leverages a pretrained text-to-image diffusion model to generate a single synthetic view from the input text and then utilizes a point cloud diffusion conditioned on the synthetic image to produce 3D point clouds. Spice-E [51] introduces a shape editing method to edit shapes semantically and transform primitive-based abstractions into highly expressive shapes. It adds structural guidance to 3D diffusion models, extending its usage beyond text-conditional generation. This approach supports a variety of applications, including 3D stylization, semantic shape editing, and text-conditional abstraction-to-3D. CLAY [52] employs a large-scale generative model designed for creating high-quality 3D assets with controllable features. It supports diverse input modalities, including text, images, point clouds, and voxels, enabling users to create intricate 3D models with minimal expertise. Cheng et al. [53] propose a generative model based on transformer architecture that creates spatially coherent and user-controllable 2D land-use layouts for virtual worlds by learning from real-world geographic data while incorporating geometric and planning objectives to enhance layout quality and adherence to user controls. DepthGAN [54] is a novel GAN-based method designed for generating accurate and geometrically consistent depth maps from 2D semantic layouts using semantically aware transformer blocks and a semantically weighted depth synthesis module, enabling controllable and effective 3D scene generation for applications like AR and VR.

DiffFacto [12] is proposed for controllable part-based 3D point cloud generation. It achieves part-level control over shapes by learning the distribution of shapes through a probabilistic generative model. The core innovation of DiffFacto lies in its ability to generate novel shapes by understanding and manipulating the factorized representation of shapes (i.e., decomposing shapes into independent parts and their configurations). This process is realized through a cross-diffusion network that simulates the complex interactions between different parts of shapes, enabling the generation of coherent and plausible 3D structures under various configurations.

Unlike DiffFacto, which controls the 3D point cloud generation through semantic segmentation labels and part transformations, we introduce StructureGraph representation that consists of part existences and part adjacency relationships, explicitly using the shape structures as user control. To enable the network to incorporate the feature information from the StructureGraph, we design the StructureGraphNet module to fuse the structural information between parts. The learned features of the StructureGraph and the part adjacency are then incorporated into the cCNF Prior module and Diffusion Transformer module to control the generation of corresponding point cloud shapes.

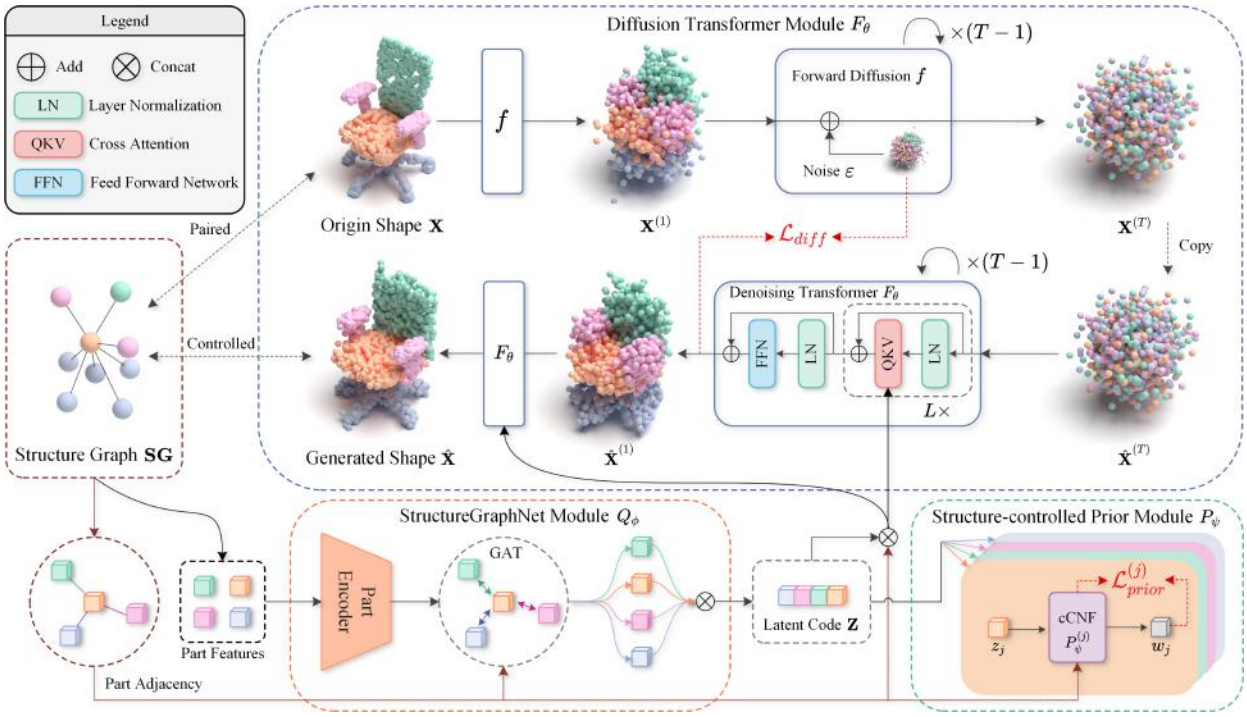

Fig. 2. Our structure-controlled 3D point cloud generation framework comprises three components: i) The StructureGraphNet Module: This module encodes part features and part adjacency of the StructureGraph into global latent code $\mathbf{Z}$. ii) The Structure-controlled Prior Module: This module utilizes cCNF models to learn the representation of the global latent code $\mathbf{Z}$ conditioned on the part adjacency of the StructureGraph. iii) The Diffusion Transformer Module: This module adds noise to 3D point cloud shapes and trains a denoising Transformer controlled by the concatenation of the global label code $\mathbf{Z}$ and the part adjacency of the StructureGraph. In particular, the part features are the segmentation labels $\mathbf{S}$ combined with the point cloud shape $\mathbf{X}$, which is only included in StructureGraph when training. The part adjacency in the figure consists of adjacency relationships $\mathbf{E}$ and part existences $\mathbf{V}$ for brevity.

3 Our Method

This section begins with an overview of the algorithm's pipeline, followed by individual introductions to each module within the framework. Finally, the complete training and sampling process is presented.

3.1 Overview

Each input point cloud shape $\mathbf{X} = \{x_i\}_{i=1}^n$ consists of n points, where each point $x_i \in \mathbb{R}^3$ represents the three-dimensional coordinates of the point. Each point cloud shape $\mathbf{X}$ of the same category is segmented into m parts, forming its semantic segmentation one-hot labels $\mathbf{S} = \{\{s_{i,j}\}_{j=1}^m\}_{i=1}^n$, where $s_{i,j}$ represents whether point x_i is assigned to the j-th part. Based on the segmentation labels $\mathbf{S}$, we introduce the StructureGraph representation, which consists of part existences $\mathbf{V} = \{v_j\}_{j=1}^m$ to denote whether part j exists in shape $\mathbf{X}$, and adjacency relationships $\mathbf{E} = \{\{e_{j,k}\}_{k=1}^m\}_{j=1}^m$ to represent whether part j is adjacent to part k, forming the StructureGraph $\mathbf{SG} = \{\mathbf{S}, \mathbf{V}, \mathbf{E}\}$. During training, the segmentation labels $\mathbf{S}$ in $\mathbf{SG}$ is also combined with the point cloud $\mathbf{X}$ as part features.

As shown in Figure 2, the overall pipeline of our method is divided into the following three modules: i) StructureGraphNet Q_ϕ: To extract structure-aware point cloud features $\mathbf{Z}$. ii) Structure-controlled cCNF Prior Module P_ψ: To learn the distribution of the extracted point cloud features $\mathbf{Z}$ conditioned on the part adjacency. iii) Structure-controlled Diffusion Transformer Module F_θ: Uses the structure-aware point cloud features $\mathbf{Z}$ extracted in i) and part adjacency as conditional context to control the generation of point cloud shapes.

Note that in Figure 2, the part features are the segmentation labels $\mathbf{S}$ combined with the point cloud shape $\mathbf{X}$, which is only included in StructureGraph when training. The part adjacency in the figure consists of adjacency relationships $\mathbf{E}$ and part existences $\mathbf{V}$ for brevity.

3.2 StructureGraphNet Module

We propose a novel StructureGraphNet (SGN) module to aggregate features of point cloud $\mathbf{X}$ and StructureGraph $\mathbf{SG}$, thereby extracting structure-aware latent features $\mathbf{Z}$. This enables subsequent generative models to generate point clouds based on the latent features and the corresponding structures.

The input of the StructureGraphNet is part features and part adjacency. As mentioned above, part features are the combination of the point cloud shape $\mathbf{X}$ and the segmentation labels $\mathbf{S}$. For each point cloud shape $\mathbf{X}$, the StructureGraphNet $Q_\phi(\mathbf{Z}|\mathbf{X}, \mathbf{S}, \mathbf{V}, \mathbf{E})$ extracts the structure-aware latent features $\mathbf{Z} = \{z_j\}_{j=1}^m$:

$$\mathbf{Z} = F_{\text{gat}}\left(F_{\text{conv}}\left(\mathbf{X}\right)^T \cdot \mathbf{S}, \mathbf{V}, \mathbf{E}\right), \tag{1}$$

where F_{conv} represents a 1D convolutional network to extract the overall feature vector $\mathbf{Z_g}$ of $\mathbf{X}$. The semantic segmentation labels $\mathbf{S}$ of $\mathbf{X}$ are multiplied by the transposed overall point cloud features $\mathbf{Z_g}^T$ to extract the local features $\mathbf{Z_l}$ of each segmented part. Each segmented part j is treated

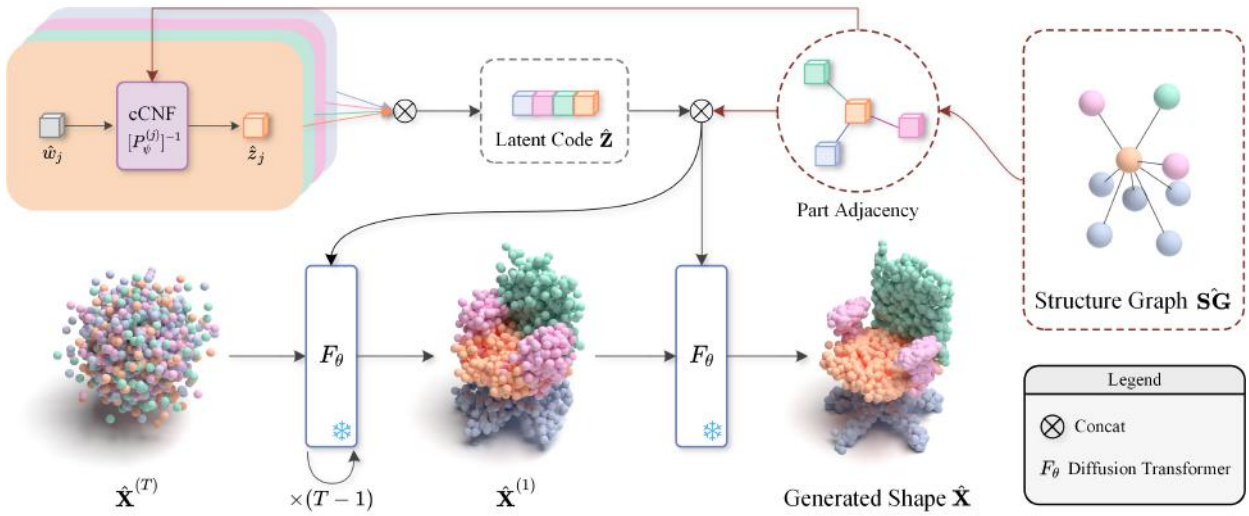

Fig. 3. Sampling process of our method. For each semantic part, the sampled noise $\hat{w}_j$ is converted to $\hat{z}_j$ through reversed cCNF, and all are combined to latent code $\hat{\mathbf{Z}}$. The user-specific StructureGraph $\hat{\mathbf{SG}}$ contains part adjacency consisting of part existences and adjacency relationships, which are then concatenated with latent code to control the diffusion Transformer F_θ gradually sampling novel and controllable point cloud $\hat{\mathbf{X}}$.

as a node in the graph, with the extracted local features $\mathbf{Z_l}$ as the node features. The part existences $\mathbf{V}$ are utilized as the part mask to ignore the node that does not exist in the graph. The adjacency relationships $\mathbf{E}$ between parts are considered as edges of the graph. A graph attention network (GAT) [55] F_{gat} is applied on this StructureGraph $\mathbf{SG}$, allowing the network to dynamically adjust the weights between individual parts and other adjacent parts, which enables increased attention to certain part relationships, such as between the armrests and seat in a chair. The final feature vector $\mathbf{Z}$ is obtained by applying the F_{gat} network on the StructureGraph $\mathbf{SG}$, which incorporates the structural information of the shape parts.

Additionally, to efficiently optimize the Evidence Lower Bound (ELBO), we use reparameterization to sample from the distribution of $Q_\phi(\mathbf{Z}|\mathbf{X},\mathbf{S},\mathbf{V},\mathbf{E})$:

$$\mathbf{Z} = \mu_\phi\left(\mathbf{X},\mathbf{S},\mathbf{V},\mathbf{E}\right) + \sigma_\phi\left(\mathbf{X},\mathbf{S},\mathbf{V},\mathbf{E}\right) * \epsilon, \tag{2}$$

where $\epsilon \sim \mathcal{N}(0,\mathbf{I})$, μ_ϕ and σ_ϕ represent the mean and standard deviation of the distribution $Q_\phi(\mathbf{Z}|\mathbf{X},\mathbf{S},\mathbf{V},\mathbf{E})$, respectively.

3.3 Structure-controlled Prior Module

In the Structure-controlled Prior Module, we employ a conditional Continuous Normalizing Flow (cCNF) [26], [27] network to learn the latent features $\mathbf{Z}$ conditioned on the part existences $\mathbf{V}$ and adjacency relationships $\mathbf{E}$. Normalizing flows are a series of reversible mappings that can transform an initial known distribution into a more complex one [9].

For each point cloud shape $\mathbf{X}$, the StructureGraphNet $Q_\phi(\mathbf{Z}|\mathbf{X},\mathbf{S},\mathbf{V},\mathbf{E})$ above extracts structure-aware feature vectors $\mathbf{Z} = \{z_j\}_{j=1}^m$. The structure-controlled cCNF Prior model $P_\psi\left(\mathbf{Z},\mathbf{V},\mathbf{E}\right) = \{P_\psi^{(j)}(z_j, v_j, e_{j.})\}_{j=1}^m$ applies a normalizing flow conditioned on the part adjacency to regularize each part feature vector z_j, yielding a transformed distribution $\mathbf{W} = \{w_j\}_{j=1}^m$:

$$w_j = P_\psi^{(j)}\left(z_j, v_j, e_{j.}\right), \tag{3}$$

where $e_{j.}$ represents all the edges adjacent to the part j.

In the sampling process, as shown in Figure 3, by inverting this normalizing flow denoted as P_ψ^{-1} and sampling $\hat{w}_j$ from a standard Gaussian distribution $\mathcal{N}(0,\mathbf{I})$, the distribution of z_j can be inversely reconstructed conditioned on the user-specific part adjacency $\hat{v}_j$ and $\hat{e}_{j.}$:

$$\hat{z}_j = [P_\psi^{(j)}]^{-1}\left(\hat{w}_j, \hat{v}_j, \hat{e}_{j.}\right) \tag{4}$$

where $\hat{z}_j$ represents the reconstructed distribution of z_j and $\hat{w}_j \sim \mathcal{N}(0,\mathbf{I})$.

Consequently, the loss function for the cCNF Prior can be derived as follows:

$$\begin{aligned}\mathcal{L}_{\text{prior}} &= \sum_{j=1}^{m} D_{\text{KL}}\left(Q_\phi\left(z_j|\mathbf{X},\mathbf{S},v_j,e_{j.}\right) || \, P_\psi^{(j)}\left(z_j,v_j,e_{j.}\right)\right) \\ &= -\sum_{j=1}^{m}\Big[\mathbb{E}_{Q_\phi(z_j|\mathbf{X},\mathbf{S},v_j,e_{j.})}[\log P_\psi^{(j)}\left(z_j,v_j,e_{j.}\right)] \\ &\quad + H\left[Q_\phi\left(z_j|\mathbf{X},\mathbf{S},v_j,e_{j.}\right)\right]\Big],\end{aligned} \tag{5}$$

where D_{KL} represents the KL divergence between two distributions and H is the entropy.

3.4 Structure-controlled Diffusion Transformer Module

In the Structure-controlled Diffusion Transformer Module, we leverage the Denoising Diffusion Probabilistic Model (DDPM) [30] to learn the conditional likelihood $P\left(\mathbf{X}|\mathbf{Z},\mathbf{S},\mathbf{V},\mathbf{E}\right)$ through an iterative denoising process.

In the forward diffusion process f, for a point cloud shape $\mathbf{X}$, Gaussian noise ε with T time steps is added, such that the distribution of the point cloud gradually becomes an independent Gaussian distribution $f(\mathbf{X}^{(t)}|\mathbf{X}^{(0)})$:

$$\begin{aligned}\mathbf{X}^{(t)} &= \sqrt{\bar{\alpha}_t}\mathbf{X}^{(0)} + \sqrt{1-\bar{\alpha}_t} * \varepsilon \\ f(\mathbf{X}^{(t)}|\mathbf{X}^{(0)}) &= \mathcal{N}(\mathbf{X}^{(t)}|\sqrt{\bar{\alpha}_t}\mathbf{X}^{(0)}, (1-\bar{\alpha}_t)\,\mathbf{I}),\end{aligned} \tag{6}$$

where $\varepsilon \sim \mathcal{N}(0,\mathbf{I})$, $t = 1,\ldots,T$, $\alpha_t = 1-\beta_t$, β_t represents the variance schedule that increases with t, $\bar{\alpha}_t = \prod_{i=1}^{t}\alpha_i$.

ALGORITHM 1: Training process of our method

Input:
Point cloud dataset $\mathbf{X}$ with semantic segmentation labels $\mathbf{S}$, part existences $\mathbf{V}$ and part adjacency relationships $\mathbf{E}$.
Output:
Trained network parameters ϕ of the SGN Q, ψ of the Prior model P, and θ of the diffusion model F.
Training process:
Step 1: Training StructureGraphNet;
for $j \leftarrow 1$ **to** m **do**
  $Q_\phi\left(z_j|\mathbf{X},\mathbf{S},v_j,e_{j.}\right) \leftarrow \mathbf{X},\mathbf{S},v_j,e_{j.}$;
  Sample $\epsilon_j \sim \mathcal{N}(0,\mathbf{I})$;
  $z_j \leftarrow \mu_\phi(\mathbf{X},\mathbf{S},v_j,e_{j.}) + \sigma_\phi(\mathbf{X},\mathbf{S},v_j,e_{j.}) * \epsilon_j$
end
Step 2: Training Prior model;
for $j \leftarrow 1$ **to** m **do**
  $P_\psi^{(j)}(z_j,v_j,e_{j.}) \leftarrow z_j,v_j,e_{j.}$;
  $\mathcal{L}_{\text{prior}} \leftarrow$
  $D_{\text{KL}}\left(Q_\phi\left(z_j|\mathbf{X},\mathbf{S},v_j,e_{j.}\right) || \, P_\psi^{(j)}\left(z_j,v_j,e_{j.}\right)\right)$.
end
Step 3: Training diffusion model.
for $i \leftarrow 1$ **to** n **do**
  Sample $t \sim \text{Uniform}\{1,\ldots,T\}$;
  Sample $\varepsilon \sim \mathcal{N}(0,\mathbf{I})$;
  $x_i^{(0)} \leftarrow x_i$;
  $x_i^{(t)} \leftarrow \sqrt{\bar{\alpha}_t}x_i^{(0)} + \sqrt{1-\bar{\alpha}_t} * \varepsilon$ (Eq. (6));
  $\mathcal{L}_{\text{diff}} \leftarrow \mathbb{E}_{\varepsilon,t,\mathbf{Z},\mathbf{S},\mathbf{V},\mathbf{E}}\left[\left\|\varepsilon - F_\theta(x_i^{(t)},\mathbf{Z},\mathbf{S},\mathbf{V},\mathbf{E})\right\|_2^2\right]$.
end

During the reverse diffusion process, the neural network F_θ is trained to model the probability distribution, using the forward process $f(\mathbf{X}^{(t-1)}|\mathbf{X}^{(t)},\mathbf{X}^{(0)})$ as an approximate posterior to approximately maximize the likelihood $\mathbb{E}_{f(\mathbf{X}^{(0)})}\log F_\theta(\mathbf{X}^{(0)})$. Based on the ELBO formula, the simplified DDPM loss function can be derived as the distance between the noise predicted by the network F_θ in the reverse diffusion and the ground truth noise ε in the forward diffusion process f:

$$\mathcal{L}_{\text{diff}} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{\varepsilon,t,\mathbf{Z},\mathbf{S},\mathbf{V},\mathbf{E}}\left[\left\|\varepsilon - F_\theta(x_i^{(t)},\mathbf{Z},\mathbf{S},\mathbf{V},\mathbf{E})\right\|_2^2\right], \tag{7}$$

where $t \sim \text{Uniform}\{1,\ldots,T\}$, $\varepsilon \sim \mathcal{N}(0,\mathbf{I})$, and $\mathbf{Z} \sim Q_\phi\left(\mathbf{Z}|\mathbf{X},\mathbf{S},\mathbf{V},\mathbf{E}\right)$. The predicted noise $F_\theta(x_i^{(t)},\mathbf{Z},\mathbf{S},\mathbf{V},\mathbf{E})$ is conditioned on the part features $\mathbf{Z}$, segmentation semantic labels $\mathbf{S}$, part existences $\mathbf{V}$, part adjacency relationships $\mathbf{E}$, and time step t. Therefore, the denoising process can be controlled through shape structure when sampling.

As shown in Figure 2, F_θ is a Transformer [41], [42] that has L cross-attention layers and a feedforward network (FFN). In each cross-attention layer:

ALGORITHM 2: Sampling process of our method

Input:
User-specific StructureGraph $\hat{\mathbf{SG}}$, which consists of segmentation labels $\hat{\mathbf{S}}$ (can be default), part existences $\hat{\mathbf{V}}$ and part adjacency relationships $\hat{\mathbf{E}}$.
Output:
Generated novel point cloud shapes $\hat{\mathbf{X}}$ with arbitrary $\hat{n}$ points.
Sampling process:
Step 1: Sampling from the Prior model;
for $j \leftarrow 1$ **to** m **do**
  Sample noise $\hat{w}_j \sim \mathcal{N}(0,\mathbf{I})$;
  $\hat{z}_j \leftarrow [P_\psi^{(j)}]^{-1}\left(\hat{w}_j,\hat{v}_j,\hat{e}_{j.}\right)$ (Eq. (4)).
end
Step 2: Sampling from the diffusion model;
Sample $\hat{\mathbf{X}}^{(T)} \sim \mathcal{N}(0,\mathbf{I})$;
for $t \leftarrow T$ **to** 1 **do**
  Sample noise $\epsilon_t \sim \mathcal{N}(0,\mathbf{I})$;
  for $i \leftarrow 1$ **to** $\hat{n}$ **do**
    $\mu_0 \leftarrow \frac{1}{\sqrt{\alpha_t}}\hat{x}_i^{(t)}$;
    $\mu_1 \leftarrow \frac{1-\alpha_t}{\sqrt{\alpha_t}\sqrt{1-\bar{\alpha}_t}}F_\theta(\hat{x}_i^{(t)},\hat{\mathbf{Z}},\hat{\mathbf{S}},\hat{\mathbf{V}},\hat{\mathbf{E}})$;
    $\sigma \leftarrow \sqrt{\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t}$;
    $\hat{x}_i^{(t-1)} = \mu_0 - \mu_1 + \sigma * \epsilon_t$ (Eq. (9)).
  end
end
$\hat{\mathbf{X}} \leftarrow \hat{\mathbf{X}}^{(0)}$.

$$\begin{aligned}
\text{CrossAttention}(Q_{ca},\, K_{ca}, V_{ca}) &= \text{Softmax}(\frac{Q_{ca}K_{ca}^T}{\sqrt{d}}) \cdot V_{ca} \\
Q_{ca} &= W_{Q_{ca}}^{(l)} \cdot \text{Concat}(\mathbf{X},\mathbf{S}) \\
K_{ca} &= W_{K_{ca}}^{(l)} \cdot \text{Concat}(\mathbf{Z},\mathbf{V},\mathbf{E},\text{Emb}(t)) \\
V_{ca} &= W_{V_{ca}}^{(l)} \cdot \text{Concat}(\mathbf{Z},\mathbf{V},\mathbf{E},\text{Emb}(t)),
\end{aligned} \tag{8}$$

where $l = 1,\ldots,L$. $W_{Q_{ca}}^{(l)}$, $W_{K_{ca}}^{(l)}$ and $W_{V_{ca}}^{(l)}$ are learnable projection matrices for the l-th cross-attention layer. $\text{Concat}(\cdot)$ is the concatenation operation, and $\text{Emb}(t)$ represents the time embedding for the time step t.

In the sampling process, as illustrated in Figure 3, given Gaussian noise as noised point cloud shape $\hat{\mathbf{X}}^{(T)} \sim \mathcal{N}(0,\mathbf{I})$ with arbitrary $\hat{n}$ points, part features $\hat{\mathbf{Z}}$, semantic segmentation labels $\hat{\mathbf{S}}$, part existences $\hat{\mathbf{V}}$, and part adjacency relationships $\hat{\mathbf{E}}$, reverse diffusion is performed from time step T to 1. Trained diffusion Transformer F_θ predicts the noise and gradually denoises $\hat{\mathbf{X}}^{(t)}$ to finally generate novel point cloud shapes $\hat{\mathbf{X}}^{(0)}$ with controllable structure. The sampling process is as follows:

$$\begin{aligned}
\hat{\mathbf{X}}^{(t-1)} = \frac{1}{\sqrt{\alpha_t}}\left(\hat{\mathbf{X}}^{(t)} - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}F_\theta(\hat{\mathbf{X}}^{(t)},\hat{\mathbf{Z}},\hat{\mathbf{S}},\hat{\mathbf{V}},\hat{\mathbf{E}})\right) \\
+ \sqrt{\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t} * \epsilon_t,
\end{aligned} \tag{9}$$

where time step $t = T,\ldots,1$ and ϵ_t is sampled noise at time step t. Note that if the semantic segmentation labels $\hat{\mathbf{S}}$ are not given, the default semantic segmentation labels

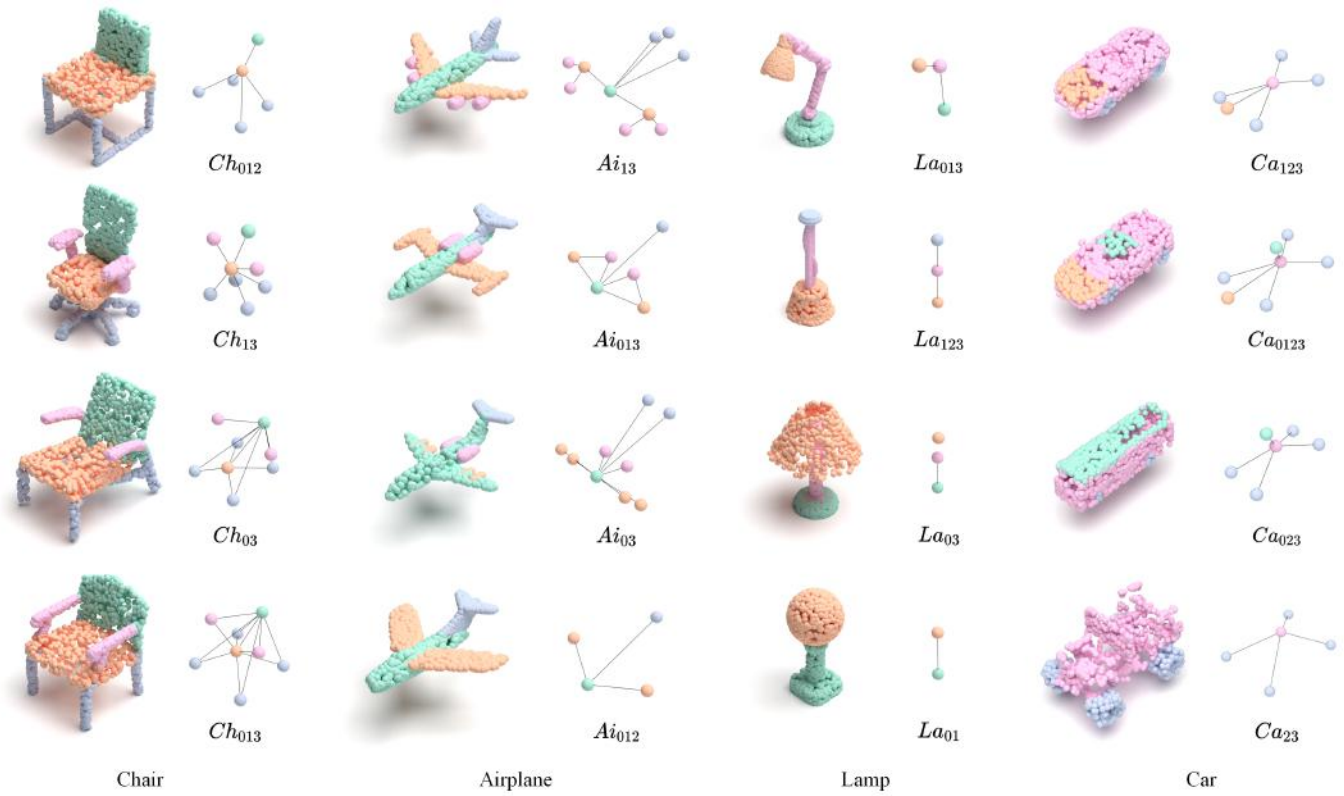

Fig. 4. 3D point cloud shapes (left column) of the ShapeNet dataset on four categories: Chair, Airplane, Lamp, and Car, with our annotated StructureGraph and their names (right column). Note that each segmented part in the StructureGraph represents only one graph node, and multiple nodes in the figure are for visualization purposes.

can be equally distributed to each point so that each existed segmentation part has the same number of points.

3.5 Training and Sampling Process

The overall training loss function $\mathcal{L}$ for our proposed method is the sum of the loss $\mathcal{L}_{\text{prior}}$ with loss weight λ for the Prior model $P_\psi(\mathbf{Z}, \mathbf{V}, \mathbf{E})$ to learn the distribution of features $\mathbf{Z}$ extracted by the StructureGraphNet $Q_\phi(\mathbf{Z}|\mathbf{X}, \mathbf{S}, \mathbf{V}, \mathbf{E})$ and the loss $\mathcal{L}_{\text{diff}}$ for the diffusion model F_θ to predict the noise:

$$\begin{aligned}\mathcal{L} &= \lambda \mathcal{L}_{\text{prior}} + \mathcal{L}_{\text{diff}} \\ &= -\lambda \sum_{j=1}^{m} \Big[\mathbb{E}_{Q_\phi(z_j|\mathbf{X},\mathbf{S},v_j,e_{j.})} [\log P_\psi^{(j)}(z_j, v_j, e_{j.})] \\ &+ H\left[Q_\phi(z_j|\mathbf{X}, \mathbf{S}, v_j, e_{j.})\right] \Big] \\ &+ \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{\varepsilon,t,\mathbf{Z},\mathbf{S},\mathbf{V},\mathbf{E}} \left[\left\| \varepsilon - F_\theta(x_i^{(t)}, \mathbf{Z}, \mathbf{S}, \mathbf{V}, \mathbf{E}) \right\|_2^2 \right]. \quad (10)\end{aligned}$$

To clarify our method, Algorithm 1 presents the training process of our approach and Algorithm 2 presents sampling process.

4 Experiments

This section first provides a detailed overview of our experiments' datasets and evaluation metrics. Then, we present the qualitative and quantitative results of our method on the widely used ShapeNet dataset and StructureNet dataset. Moreover, we compare our method with the results of existing state-of-the-art 3D point cloud generation methods. We also perform some ablation studies to verify the effectiveness of each module in our method. We also show the results of reconstructing surfaces from the generated point clouds. Finally, the implementation details and performance are presented.

4.1 Experimental Datasets

4.1.1 ShapeNet Dataset

Following prior works [9], [10], [12], the datasets used in this paper are four categories of the ShapeNet dataset [40]: Chair, Airplane, Lamp, and Car, with part segmentation labels [56]. Each category contains 3053, 2349, 1261, 740 training shapes and 704, 341, 286, 158 test shapes, respectively. On each ShapeNet category, we train our model separately and generate novel point cloud shapes.

Since the dataset only provides segmentation labels for each shape category, instead of simply computing the minimum distance between two segmented parts to obtain the part adjacency, we manually annotate the adjacency relationships between the segmented parts of the 3D point cloud shapes to construct the StructureGraph. As shown in Figure 4, for each category, the left column is the 3D point cloud shapes with segmentation labels, and the right column is the StructureGraph we annotated along with their names. For example, Ch_{13} indicates that the 1-th part (seat) and the 3-th part (armrest) of this chair are adjacent (we assume that the chair's seat is adjacent to the backrest

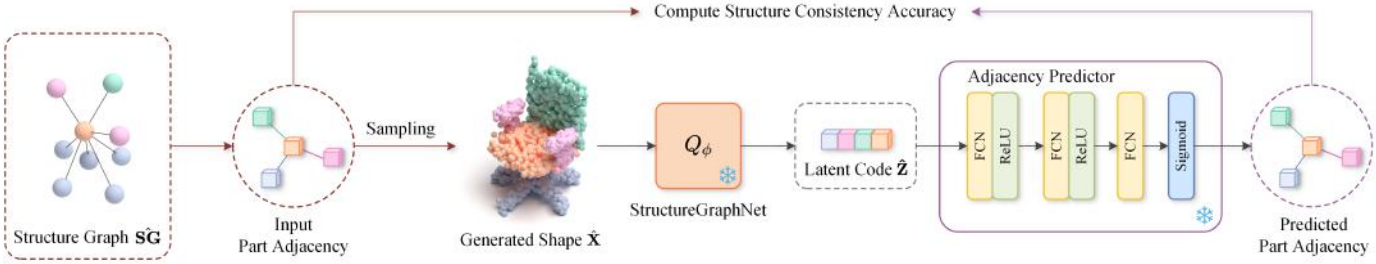

Fig. 5. Computing process of our proposed Structure Consistency Accuracy between input part adjacency and predicted part adjacency. FCN in the AdjacencyPredictor represents the fully connected neural networks.

and leg for brevity). Note that each segmented part in the StructureGraph represents only one graph node, and multiple nodes in the figure are for visualization purposes. We randomly sample 2048 points from each of the point cloud shape and normalize each of them to zero mean and unit variance.

Moreover, we calculate the Cyclomatic Complexity [57] to analyze the complexity of the shape's structure. There are 171 chairs that have reached the maximum complexity of a 4-node complete graph: 10. We also counted that there were a total of 2087 chairs with a complexity of 5 or more. This indicates that the structure of most chairs in the 4-node graph has reached a relatively complex level.

4.1.2 StructureNet Dataset

StructureNet [20] represents shapes as n-ary part hierarchies with adjacency between sibling parts, they compute the minimum distance between two segmented parts to obtain the part adjacency on the PartNet [22] dataset. The shape structure in the StructureNet dataset is relatively complex.

To adapt the structure of StructureNet dataset to apply in our method, we only use the first level of the root node of the StructureNet data. We trained our method separately on four categories of the StructureNet dataset: Chair, Vase, TrashCan, and Bed. We randomly split the dataset into 85% training set and 15% testing set.

4.2 Evaluation Metrics

We employ two types of metrics to evaluate our proposed structure-controlled 3D point cloud generation method: Shape Generation Metrics and Structure Consistency Accuracy. The Shape Generation Metrics are used to assess the realism and diversity of the generated point cloud shapes while the Structure Consistency Accuracy is utilized to evaluate the consistency between the generated point cloud shapes and the input shape structure.

Shape Generation Metrics. Following prior works, we use the Chamfer Distance (CD) [43] and the Earth Mover's Distance (EMD) [43] to evaluate the reconstruction quality of the generated point cloud shape. To evaluate the generation quality, we employ the coverage score (COV) [43], the minimum matching distance (MMD) [43], the Jenson-Shannon divergence (JSD) [43], and 1-NN classifier accuracy (1-NNA) [9]. MMD is a metric for evaluating the quality of generated point clouds. For each point cloud in the reference set, the distance to its nearest neighbor in the generated set is computed and averaged. COV measures the proportion of point clouds in the reference set that match at least one point cloud in the generated set. For each point cloud in the generated set, its nearest neighbor in the reference set is marked as matched. 1-NNA is used for two-sample tests to assess whether two distributions are identical. JSD is used to calculate the distance between marginal point distributions.

Structure Consistency Accuracy. The core of the structure-controllable generation network proposed in this paper lies in generating point cloud shapes that conform to the input structure. Therefore, in addition to measuring the quality of the generated point cloud shapes, it is also necessary to measure the consistency between the user-input StructureGraph $\hat{\mathbf{SG}} = \{\hat{\mathbf{S}}, \hat{\mathbf{V}}, \hat{\mathbf{E}}\}$ and the generated point cloud shapes $\hat{\mathbf{X}}$. We propose the Structure Consistency Accuracy (SCA) to evaluate this consistency with the following formula:

$$\begin{aligned} \mathrm{SCA}(\hat{\mathbf{X}}, \hat{\mathbf{V}}, \hat{\mathbf{E}}) &= \frac{\sum_{\hat{v}_j \in \hat{\mathbf{V}}} \sum_{\hat{e}_{j,k} \in \hat{\mathbf{E}}} [q_v * q_e]}{m^2} \\ q_v &= \mathbb{I}\left[\hat{v}_j, [N_v(\hat{\mathbf{X}})]_j\right] \\ q_e &= \mathbb{I}\left[\hat{e}_{j,k}, [N_e(\hat{\mathbf{X}})]_{j,k}\right], \end{aligned} \tag{11}$$

where $j = 1, \ldots, m$, $k = 1, \ldots, m$, and $N(\cdot)$ represents the neural networks that predict the StructureGraph of the generated point cloud $\hat{\mathbf{X}}$. Therefore, $[N_v(\cdot)]_j$ predicts whether the j-th part exists while $[N_e(\cdot)]_{j,k}$ predicts whether the j-th part is adjacent to the k-th part. As illustrated in Figure 5, the neural networks $N(\cdot)$ contain frozen StructureGraphNet and AdjacencyPredictor, which are also trained using the point cloud shapes with their corresponding StructureGraph on each category of the ShapeNet dataset. Similarly, $\hat{v}_j$ is whether the j-th part exists and $\hat{e}_{j,k}$ is whether the j-th part is adjacent to the k-th part in the input StructureGraph. $\mathbb{I}[a, b]$ is an indicator function that equals 1 if $a = b$ and 0 otherwise. This metric measures the consistency between the input StructureGraph and the predicted StructureGraph of the generated shape.

4.3 Experimental Results and Comparisons

In our work, we evaluate the effectiveness of our proposed method by comparing it with various controllable 3D point cloud generation methods and pre-trained 3D shape generation methods on the ShapeNet dataset. We also evaluate the performance of our method on four categories of the StructureNet dataset.

TABLE 1
Comparing our method with 3D shape generation methods on four categories of the ShapeNet dataset in Shape Generation Metrics. MMD scores and JSD scores are multiplied by 10^2. COV scores and 1-NNA scores are reported in %.

		MMD (↓)		COV (%, ↑)		1-NNA (%, ↓)		JSD (↓)
Shape	Model	CD	EMD	CD	EMD	CD	EMD	-
Chair	PointFlow [9]	9.00	35.37	22.67	25.87	93.20	96.67	5.02
	DPM [10]	8.89	39.77	29.60	19.20	88.13	95.87	4.39
	DiffFacto [12]	6.28	32.83	45.37	**39.13**	75.48	**88.54**	2.75
	SPAGHETTI [47]	10.64	30.37	43.37	32.67	87.74	93.02	3.91
	SALAD [48]	10.60	**30.21**	45.07	36.24	85.16	90.08	3.72
	StrucADT (Ours)	**6.26**	30.92	**48.80**	33.87	**69.87**	93.60	**1.46**
Airplane	PointFlow [9]	3.42	23.76	35.21	34.44	87.04	90.74	2.27
	DPM [10]	2.94	25.28	37.41	22.59	77.59	95.00	1.73
	DiffFacto [12]	2.81	22.08	40.20	35.34	76.23	**89.07**	1.62
	SPAGHETTI [47]	5.24	25.29	45.56	35.93	89.13	90.78	6.52
	SALAD [48]	4.12	**21.29**	**47.04**	33.34	84.97	90.12	6.27
	StrucADT (Ours)	**2.69**	23.00	42.59	**37.04**	**74.81**	94.07	**1.60**
Lamp	PointFlow [9]	10.82	42.09	43.87	41.29	82.26	88.06	6.74
	DPM [10]	11.32	40.41	43.23	38.06	80.32	93.23	4.87
	DiffFacto [12]	9.33	32.74	47.88	**48.62**	71.40	79.21	3.44
	StrucADT (Ours)	**8.11**	**31.49**	**49.03**	45.81	**61.29**	**68.06**	**2.41**
Car	PointFlow [9]	3.80	**21.78**	52.75	39.56	71.98	**76.92**	1.88
	DPM [10]	3.54	23.37	51.65	23.08	60.44	90.66	1.86
	DiffFacto [12]	**3.12**	22.74	55.09	43.11	62.17	80.10	1.90
	StrucADT (Ours)	3.34	21.93	**57.14**	**43.96**	**58.24**	79.67	**1.84**

TABLE 2
Evaluating the Structure Consistency Accuracy of our method on the Chair category. SCA scores are reported in %.

Shape	Model	SCA-Ch_{012}	SCA-Ch_{03}	SCA-Ch_{13}	SCA-Ch_{23}	SCA-Ch_{013}	SCA-Ch_{023}	SCA-Ch_{123}	SCA-Ch_{0123}
Chair	PointFlow [9]	90.17	82.78	80.23	81.12	72.58	73.98	72.45	63.90
	DPM [10]	93.88	82.00	81.75	83.12	73.75	73.37	73.00	63.37
	DiffFacto [12]	95.42	85.62	82.45	**85.10**	80.98	81.12	76.40	74.23
	StrucADT (Ours)	**97.96**	**91.20**	**85.20**	82.65	**87.24**	**83.55**	**77.68**	**77.30**

TABLE 3
Evaluating the Structure Consistency Accuracy of our method on the Airplane category. SCA scores are reported in %.

Shape	Model	SCA-Ai_{012}	SCA-Ai_{03}	SCA-Ai_{13}	SCA-Ai_{23}	SCA-Ai_{013}	SCA-Ai_{023}	SCA-Ai_{123}	SCA-Ai_{0123}
Airplane	PointFlow [9]	**85.46**	78.70	77.42	75.26	71.17	69.13	69.39	61.35
	DPM [10]	83.16	77.81	77.81	77.68	69.39	70.41	69.13	63.01
	DiffFacto [12]	81.22	82.16	80.73	**79.18**	83.90	75.02	71.74	73.95
	StrucADT (Ours)	83.04	**86.22**	**94.26**	70.92	**93.75**	**76.79**	**80.36**	**84.06**

TABLE 4
Evaluating the Structure Consistency Accuracy of our method on the Lamp category. SCA scores are reported in %.

Shape	Model	SCA-La_{012}	SCA-La_{03}	SCA-La_{13}	SCA-La_{23}	SCA-La_{013}	SCA-La_{023}	SCA-La_{123}	SCA-La_{0123}
Lamp	PointFlow [9]	77.81	71.30	72.96	71.94	66.96	69.13	68.24	60.71
	DPM [10]	79.08	72.83	74.11	72.19	67.09	68.75	66.71	63.01
	DiffFacto [12]	78.94	74.09	**80.17**	76.92	70.60	71.12	72.58	75.50
	StrucADT (Ours)	**81.51**	**77.81**	78.44	**78.95**	**78.95**	**77.30**	**76.91**	**79.85**

TABLE 5
Evaluating the Structure Consistency Accuracy of our method on the Car category. SCA scores are reported in %.

Shape	Model	SCA-Ca_{03}	SCA-Ca_{13}	SCA-Ca_{23}	SCA-Ca_{013}	SCA-Ca_{023}	SCA-Ca_{123}	SCA-Ca_{0123}
Car	PointFlow [9]	65.43	65.82	65.18	62.37	65.94	64.27	61.35
	DPM [10]	65.31	68.37	67.60	64.54	63.78	67.60	63.39
	DiffFacto [12]	70.92	**76.19**	71.37	75.52	77.61	74.09	81.62
	StrucADT (Ours)	**76.02**	74.62	**76.53**	**86.09**	**88.65**	**86.85**	**98.98**

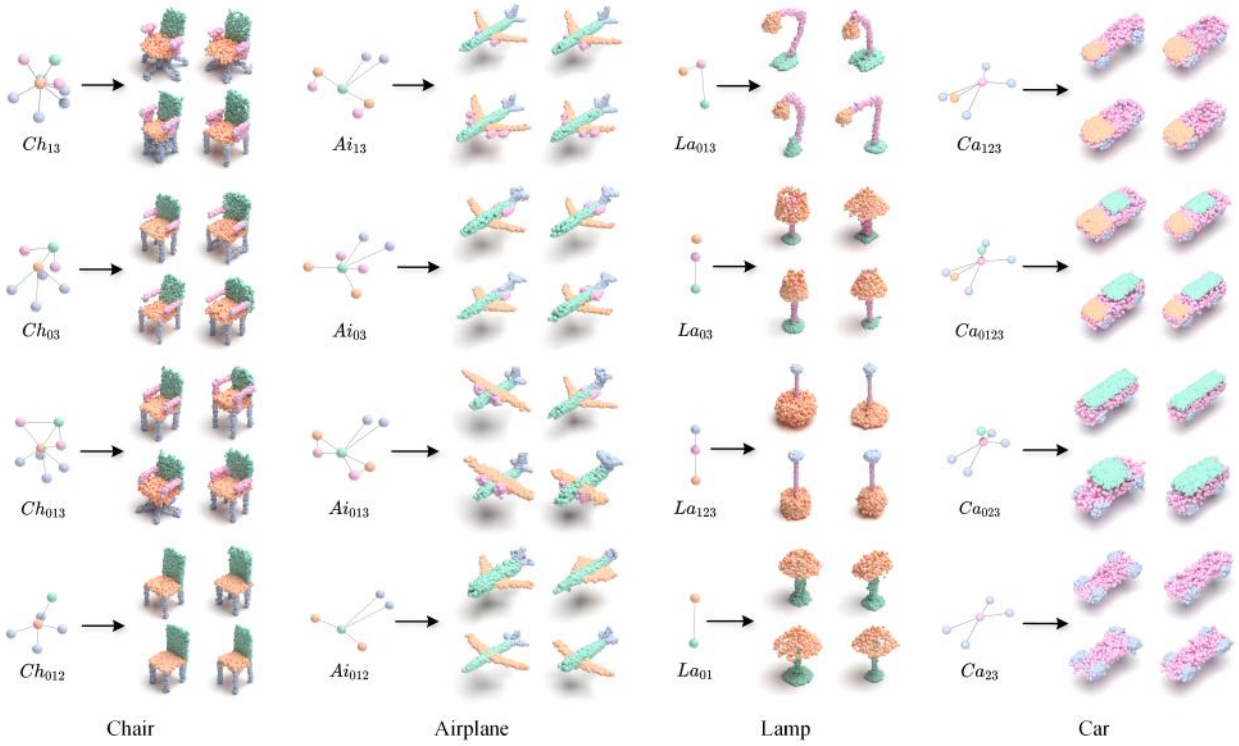

Fig. 6. Generated 3D point cloud shapes (right column) controlled by the user-specific StructureGraph (left column) on four categories of the ShapeNet dataset: Chair, Airplane, Lamp, and Car.

Fig. 7. Comparison results of generated 3D point cloud shapes between our method (StrucADT) with pre-trained 3D shape generation methods: SPAGHETTI and SALAD on the Chair and Airplane categories of the ShapeNet dataset.

4.3.1 *ShapeNet Dataset*

Following DiffFacto [12], to ensure a fair comparison, we adapt PointFlow [9], DPM [10], and DiffFacto [12] to be controlled by our proposed StructureGraph representation so that we can compare with these methods in Shape Generation Metrics and Structure Consistency Accuracy to evaluate both the generation quality and structure controllability.

Comparison in Shape Generation Metrics. When comparing in Shape Generation Metrics, we evaluate all the methods on the test shapes of the ShapeNet dataset with our annotated StructureGraph, as illustrated in Figure 4. As shown in Table 1, we compare our method with structure-adapted PointFlow, DPM, and DiffFacto in Shape Generation Metrics on the four categories of the ShapeNet dataset. MMD and JSD scores are multiplied by 10^2. COV scores and 1-NNA scores are reported in %. Based on the data in Table 1, our proposed StrucADT attained higher scores in Shape Generation Metrics compared to other methods on each category of the ShapeNet dataset, proving that our method can produce high-quality and varying point clouds controlled by the structure of test shapes. In some metrics of each category, our method is slightly lower than PointFlow and DiffFacto, but our method is generally better than the other three methods on each category. Our experimental results demonstrate that our structure-controlled 3D point cloud generation method outperforms other methods in generating novel and diverse 3D point cloud shapes.

We also compare our method with pre-trained 3D shape generation methods: SPAGHETTI [47] and SALAD [48]. We used the pre-trained models of the SPAGHETTI and

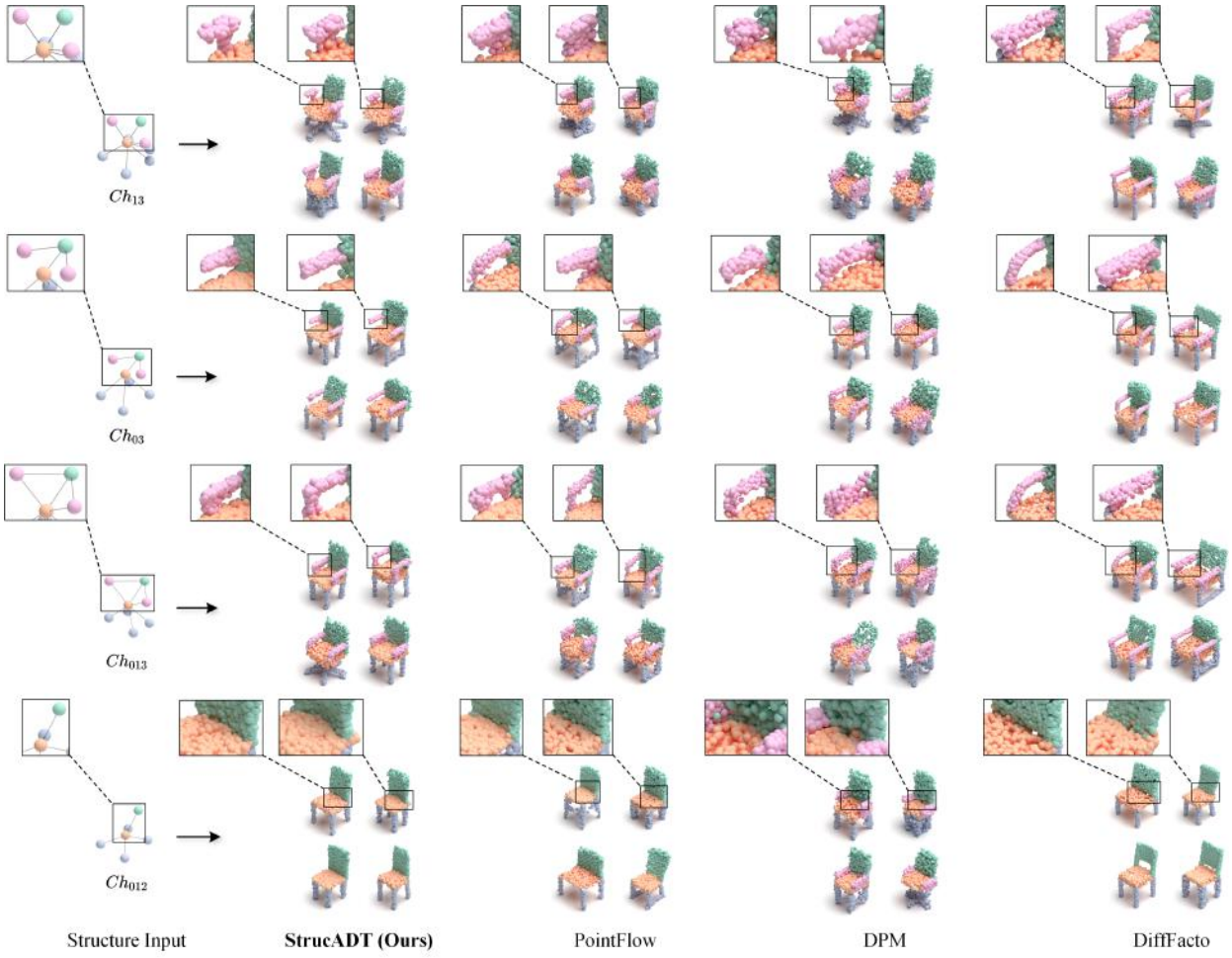

Fig. 8. Comparison results of generated 3D point cloud shapes controlled by the user-specific StructureGraph on the Chair category of the ShapeNet dataset. The enlarged part of the figure shows the changes in the part adjacency relationships corresponding to different input structures of the chair. Our method can generate shapes consistent with the input structures, while other methods fail.

SALAD, and sampled 3D shapes to evaluate the Shape Generation Metrics on the Chair and Airplane categories. As shown in Table 1 and Figure 7, our method outperforms SPAGHETTI and is comparable to SALAD in generating high-quality and diverse point clouds.

Comparison in Structure Consistency Accuracy. When comparing in Structure Consistency Accuracy, the input StructureGraph consists of the user-specific part existences, part adjacency relationships, and the default semantic segmentation labels that give each existing segmentation part the same number of points. We generate 50 shapes for one input StructureGraph on each category of the ShapeNet dataset to compute the averaged Structure Consistency Accuracy. Although our method can generate point clouds with arbitrary points, the generated shapes still contain 2048 points, the same as the input shapes. Figure 6 displays the generation results of our proposed StrucADT controlled by the user-specific input StructureGraph on the four categories of the ShapeNet dataset. For each category, the left column is the input structures to control point cloud generation, while the right column is the four generated shapes controlled by each input structure. All our generated point cloud shapes are consistent with the input structures on the four categories.

As shown in Table 2 to Table 5, we compare our method with structure-adapted PointFlow, DPM, and DiffFacto in Structure Consistency Accuracy on each category of the ShapeNet dataset. SCA scores are reported in %. We experiment with all the structures occurring on the four categories of the ShapeNet dataset, where the structures with the highest occurrences are listed in the table. For example, SCA-Ch_{03} represents the Structure Consistency Accuracy of the generated chair with the 3-th part (armrests) only adjacent to the 0-th part (back), and SCA-Ch_{012} represents the Structure Consistency Accuracy of the generated chair with no 3-th part (armrests). Note that Ca_{012} does not exist in the Car category because all cars have the 3-th part (body). Based on the data in Table 2 to Table 5, our proposed StrucADT achieves higher scores in Structure Consistency Accuracy compared to other methods on each category of the ShapeNet dataset. In addition, we find that the Structure Consistency Accuracy decreases as the part adjacency of the input structure increases (e.g., from Ch_{012} to Ch_{0123}) on the Chair category, indicating that more complex structures are more challenging to control the generation of the corresponding 3D point cloud shapes. It is necessary to balance the quality of point cloud generation and structural consistency. Therefore, our method may not achieve the best quality performance of the point clouds generated in some

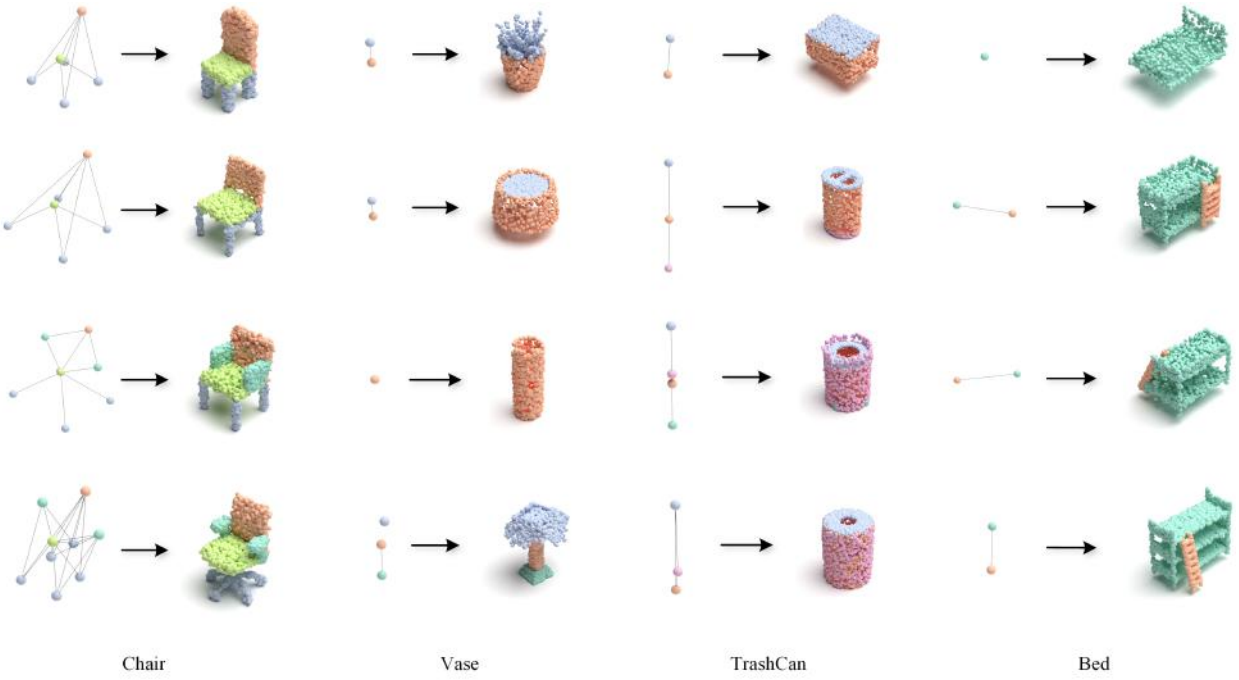

Fig. 9. Generated 3D point cloud shapes (right column) controlled by the user-specific StructureGraph (left column) on four categories of the StructureNet dataset: Chair, Vase, TrashCan, and Bed.

TABLE 6
Evaluating the performance of our method on four categories of the StructureNet dataset in Shape Generation Metrics. MMD scores and JSD scores are multiplied by 10^2. COV scores and 1-NNA scores are reported in %.

	MMD (↓)		COV (%, ↑)		1-NNA (%, ↓)		JSD (↓)
Shape	CD	EMD	CD	EMD	CD	EMD	-
Chair	5.95	31.71	40.29	34.07	75.95	92.98	1.13
Vase	8.43	31.78	53.45	48.28	51.29	60.78	1.84
TrashCan	5.96	9.07	58.97	48.72	47.44	73.08	2.00
Bed	10.79	38.38	50.00	46.15	42.31	53.85	5.14

cases, but it can generate point clouds corresponding to the structure input by the user.

Figure 8 shows a qualitative comparison of our method with the other three methods on the chair category of the ShapeNet dataset. The enlarged part of the figure shows the changes in the part adjacency relationships corresponding to different input structures of the chair. Our method can generate shapes that are consistent with the input structures. Especially when the input structures are Ch_{13} and Ch_{03}, our method can generate chairs whose armrests are only attached to the seat and whose armrests are only attached to the back, respectively, while other methods fail to generate these shapes. Our experimental results demonstrate that our 3D point cloud generation method can generate shapes controlled by the input structures, outperforming other methods and achieving better structure controllability.

4.3.2 StructureNet Dataset

We also evaluate the performance of our method on four categories of the StructureNet dataset in Shape Generation Metrics, as shown in Table 6. Table 6 and Figure 9 indicate that our method can generate high-quality point cloud shapes consistent with the input structure on the relatively complex StructureNet dataset.

It is worth noting that the StructureGraph encodes only abstract part existences and adjacency relationships, without spatial location or orientation. Hence, the diversity in part layouts seen in the last column of Figure 9 arises solely from the stochastic nature of the diffusion-based generation process. This design allows our model to flexibly generate multiple plausible spatial configurations that are consistent with the same abstract structure.

Moreover, our method naturally supports disconnected nodes in the input StructureGraph, i.e., parts without any adjacent connections. These nodes are encoded individually in the StructureGraphNet without receiving messages from neighbors, and their spatial placement during generation is determined stochastically. For example, in the last Vase instance in Figure 9, the top part is structurally disconnected from the body, and our model successfully generates it as a separate, non-adjacent component, demonstrating correct handling of isolated parts under structure control.

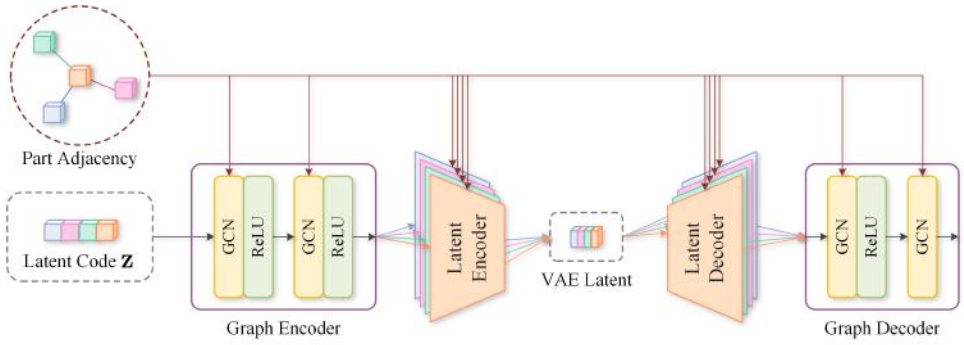

Fig. 10. Network architecture of GraphCVAE in the ablation experiments of the Prior module. GCN in the Graph Encoder and Graph Decoder represents the graph convolutional neural networks.

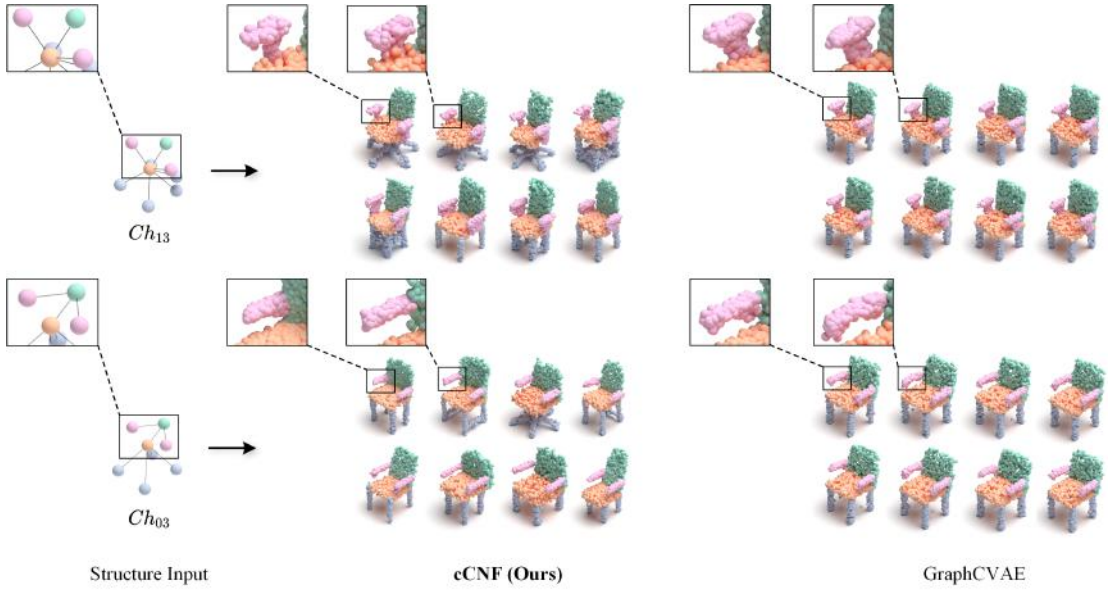

Fig. 11. Comparison results of ablation experiments on network framework of prior module controlled by the user-specific StructureGraph on chair category. Compared to GraphCVAE, our method generates shapes consistent with the input structures and ensures the quality and diversity of the generated shapes. In contrast, GraphCVAE can only generate similar and repetitive shapes with excessive control.

4.4 Ablation Study

In this section, we conduct ablation experiments on the critical modules of our method and evaluate the structure generalization of our method. The other modules remain unchanged when one module is ablated in the experiments.

4.4.1 StructureGraphNet

To validate the contributions of our proposed StructureGraphNet (SGN) to the shape generation's effectiveness, we ablate our StructureGraphNet with the PointNet encoder [58], which only takes in each shape part without part adjacency. Table 7 shows that our proposed StructureGraphNet outperforms the PointNet encoder in Shape Generation Metrics, and Table 8 displays that our proposed StructureGraphNet achieves higher Structure Consistency Accuracy compared to the PointNet encoder, proving the effectiveness of the StructureGraphNet in extracting structure-aware latent features.

4.4.2 cCNF Prior

We ablate our cCNF Prior with our proposed StructureGraph conditioned VAE (GraphCVAE) Prior, which encodes the latent code $\mathbf{Z}$ and decodes it conditioned on the part adjacency with reconstruction loss and latent regularization loss. Figure 10 displays the network architecture of GraphCVAE. As shown in Figure 11, GraphCVAE has more controllability on point cloud generation conditioned on the StructureGraph, but the generated shapes have low diversity and quality. Table 7 illustrates that our cCNF Prior outperforms the GraphCVAE Prior, achieving higher shape generation scores. Table 8 shows that GraphCVAE Prior has higher Structure Consistency Accuracy than cCNF Prior. Compared to GraphCVAE, our method generates shapes consistent with the input structures and ensures the quality and diversity of the generated shapes. In contrast, GraphCVAE can only generate similar and repetitive shapes with excessive control. Therefore, our proposed cCNF Prior is superior to the GraphCVAE Prior.

TABLE 7
Ablation experiments on the network frameworks of the StructureGraphNet module, Prior module, Diffusion Transformer module and our overall framework using the Chair category in Shape Generation Metrics. SGN is the abbreviation of our proposed StructureGraphNet. DiT is the abbreviation of our Diffusion Transformer module. MMD scores and JSD scores are multiplied by 10^2. COV scores and 1-NNA scores are reported in %.

		MMD (↓)		COV (%, ↑)		1-NNA (%, ↓)		JSD (↓)
Shape	Model	CD	EMD	CD	EMD	CD	EMD	-
Chair	PointNet [58]	**6.23**	32.38	48.00	**36.00**	71.73	96.13	2.08
	SGN (Ours)	6.26	**30.92**	**48.80**	33.87	**69.87**	**93.60**	**1.46**
Chair	GraphCVAE	10.28	36.18	29.33	25.87	93.47	98.27	8.49
	cCNF (Ours)	**6.26**	**30.92**	**48.80**	**33.87**	**69.87**	**93.60**	**1.46**
Chair	SGN+PointFlow [9]	**6.15**	31.83	38.54	**35.93**	71.47	95.07	1.91
	SGN+DPM [10]	6.51	34.53	39.73	21.07	80.53	96.13	2.39
	SGN+DiT (Ours)	6.26	**30.92**	**48.80**	33.87	**69.87**	**93.60**	**1.46**
Chair	Two Diffusion Models [42]	7.30	33.37	42.40	27.73	81.33	97.20	4.76
	Single VAE [59]	9.34	43.75	32.53	20.99	85.74	95.56	4.71
	StrucADT (Ours)	**6.26**	**30.92**	**48.80**	**33.87**	**69.87**	**93.60**	**1.46**

TABLE 8
Ablation experiments on the network frameworks of the StructureGraphNet module, Prior module and Diffusion Transformer module using the Chair category in Structure Consistency Accuracy. SGN is the abbreviation of our proposed StructureGraphNet. DiT is the abbreviation of our Diffusion Transformer module. SCA scores are reported in %.

Shape	Model	SCA-Ch_{012}	SCA-Ch_{03}	SCA-Ch_{13}	SCA-Ch_{23}	SCA-Ch_{013}	SCA-Ch_{023}	SCA-Ch_{123}	SCA-Ch_{0123}
Chair	PointNet [58]	97.83	**91.33**	84.21	**83.80**	86.27	83.41	76.95	75.81
	SGN (Ours)	**97.96**	91.20	**85.20**	82.65	**87.24**	**83.55**	**77.68**	**77.30**
Chair	GraphCVAE	97.83	**92.09**	**93.24**	**84.69**	**92.47**	**85.46**	72.96	**79.72**
	cCNF (Ours)	**97.96**	91.20	85.20	82.65	87.24	83.55	**77.68**	77.30
Chair	SGN+PointFlow [9]	97.83	**91.63**	85.13	82.25	86.25	82.37	**79.25**	76.37
	SGN+DPM [10]	96.94	88.14	85.01	79.72	85.46	79.46	78.32	76.53
	SGN+DiT (Ours)	**97.96**	91.20	**85.20**	**82.65**	**87.24**	**83.55**	77.68	**77.30**

TABLE 9
Evaluating the structure generalization of our method on the Chair category of the ShapeNet dataset in Shape Generation Metrics under different Structure Ratios. Structure Ratios refer to the percentage of training samples containing the target structure Ch_{013}. MMD scores and JSD scores are multiplied by 10^2. COV scores and 1-NNA scores are reported in %.

		MMD (↓)		COV (%, ↑)		1-NNA (%, ↓)		JSD (↓)
Shape	Structure Ratios	CD	EMD	CD	EMD	CD	EMD	-
Chair	0%	7.40	33.25	40.06	32.81	79.81	94.48	2.54
	10%	7.39	34.16	41.96	33.22	79.20	94.23	2.91
	20%	7.59	32.79	42.13	32.16	78.94	96.06	2.16
	30%	7.17	31.72	42.29	34.94	78.38	**92.34**	2.12
	50%	6.96	31.84	43.13	**35.22**	77.99	93.51	2.05
	85% (Full)	**6.26**	**30.92**	**48.80**	33.87	**69.87**	93.60	**1.46**

TABLE 10
Evaluating the structure generalization of our method on the Chair category in Structure Consistency Accuracy under different Structure Ratios. Structure Ratios refer to the percentage of training samples containing the target structure Ch_{013}. SCA scores are reported in %.

Shape	Structure Ratios	SCA-Ch_{013}
Chair	0%	80.38
	10%	82.62
	20%	84.99
	30%	85.87
	50%	86.49
	85% (Full)	**87.24**

4.4.3 Diffusion Transformer

We also ablate our Diffusion Transformer (DiT) with PointFlow [9] and DPM [10]. SGN+PointFlow utilizes another CNF module controlled by the latent code and part adjacency to learn the distribution of the origin point cloud. SGN+DPM uses DDPM to diffuse point clouds, which is also controlled by the latent code and part adjacency. The denoising networks used in SGN+DPM are a series of fully connected layers with feature concatenation and squashing. Table 7 and Table 8 show that our SGN+DiT outperforms SGN+PointFlow and SGN+DPM module in both Shape Generation Metrics and Structure Consistency Accuracy, proving the effectiveness of our proposed part adjacency conditioned diffusion Transformer on structure controllable point cloud generation.

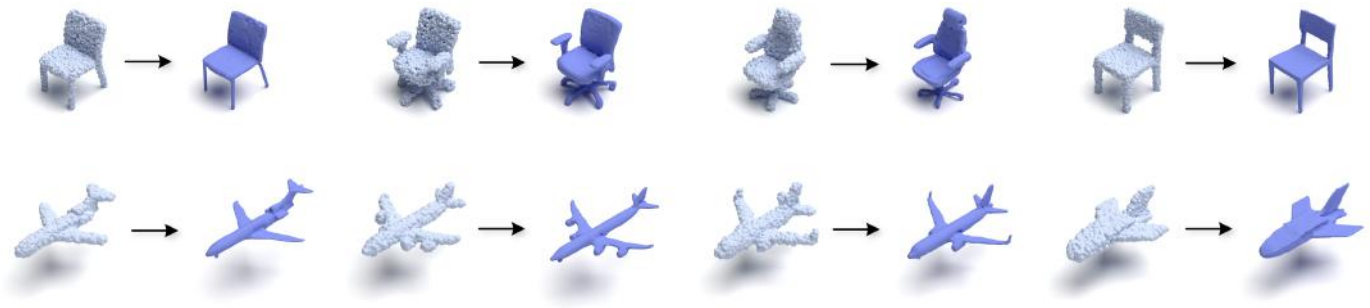

Fig. 12. The results of reconstructing surfaces from the point clouds generated on the Chair and Airplane categories.

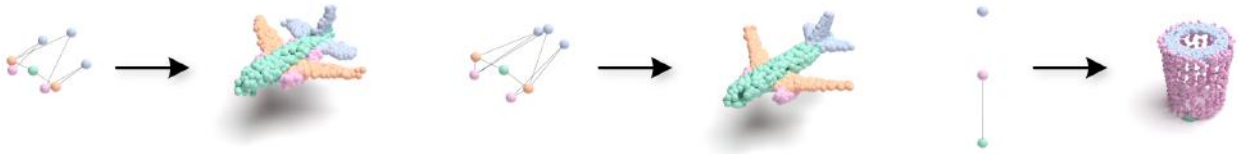

Fig. 13. When the input structure is very different from the training set, our method has a specific generalization ability, as the input aircraft engine is to be adjacent to both the wing and the tail wing (left column). Still, it may ignore this input structure, and the generated aircraft engine is only adjacent to the wing (middle column). Besides, the top of the generated trash can is still adjacent to the body, although the two parts are not connected in the input structure graph (right column).

4.4.4 Overall Framework

We also performed ablation studies on our overall framework. We compare our overall framework with the framework using two diffusion models and the framework using a single VAE. For the framework of two diffusion models (DiT [42]), one diffusion model is used to learn latent variables and the other one is used for point cloud generation. For the framework of a single VAE, we utilize the Conditional VAE [59]. As shown in Table 7, our method outperforms the framework using two diffusion models and the framework using a single VAE.

4.4.5 Structure Generalization

We designed an ablation experiment to evaluate the effectiveness of our method in structure generalization. On the Chair category of the ShapeNet dataset, we regard structure Ch_{013} (the chair's armrest is connected to both the seat and the backrest) as the novel structure and use it as the test set, and other structures as training sets. To assess the generalization ability, we vary the proportion of samples with the structure Ch_{013} included in the training set, which we refer to as the Structure Ratios. We experiment with Structure Ratios of 0%, 10%, 20%, 30%, 50%, and 85%, where 85% corresponds to the original full training setup. As shown in Table 9 and Table 10, when the Structure Ratios is 0% (i.e., Ch_{013} is entirely unseen during training), our method is still able to generate shapes that are reasonably consistent with the desired structure, indicating its ability to generalize to novel structural configurations. As more structures Ch_{013} are added to the training set, both the structure consistency accuracy and the quality of the generated point clouds gradually improve. These results demonstrate that our method exhibits a strong capacity for generalizing to unseen or rare structures by leveraging structural priors learned from other configurations.

4.5 Surface Reconstruction Application

Figure 12 shows the results of reconstructing surfaces from the point clouds generated on the Chair and Airplane categories. We use the pretrained SDF regression model of the Point-E [11] to produce meshes from point clouds.

We follow the previous work, DiffFacto [12], and randomly downsample the point clouds with more than 2048 points in the point cloud dataset to 2048 points. Therefore, the point clouds generated by random sampling are relatively challenging to reconstruct into meshes. In order to be more suitable for surface reconstruction, we provide the results of the point clouds generated by the dataset with the farthest point sampling, as well as the meshes reconstructed on these point clouds. Figure 12 indicates that the point cloud shapes we generated can be well reconstructed as meshes.

4.6 Implementation Details and Performance

We implement the proposed algorithm using Python. We use the public pytorch implementation of DDPM as the foundation for our point cloud generation model, which is trained using a batch size of 128 and an initial learning rate of 0.001 for 100,000 iterations.

Our approach is trained and tested on a PC with an Intel Core i7 CPU, 32GB of RAM, and an NVIDIA GeForce GTX 4090 GPU. Our 3D point cloud generation method consists of two main phases, including the training phase and the sampling phase. In the training phase, our algorithm takes about 2 days to train a single category shape for 100,000 iterations. In the sampling phase, generating 50 structure-controlled 3D point cloud shapes takes about 4 seconds.

5 Limitations and Future Work

The shape generation method presented in this paper requires the annotation of segmentation semantic labels for

3D point cloud shapes and the connectivity relationships between segmented parts. A potential follow-up work to this paper is self-supervised controllable 3D point cloud shape generation. This approach learns through self-supervision to use the shape itself as the control condition without the need for additional structural information. This could address the issue of annotation being time-consuming and labor-intensive.

We admit that our method relies heavily on the structures that appear in the training set. The structure of the test input in the above ablation study is relatively similar to that in the training set, so our method can have a specific generalization ability. However, when the input structure is very different from the training set, as shown in Figure 13, our method may ignore the very different input structure and generate shapes based on the learned structure that is most similar to the input structure. We believe that increasing the data in the dataset or fine-tuning on a specific dataset can alleviate this problem, just like Stable Diffusion [35] and Large Language Models [60].

Another possible future direction is combining 3D shape generation with text control information to achieve text-controlled 3D point cloud shape generation. While this paper has realized controllable generation based on 3D shape structure, text is a more abstract representation than shape structure. Realizing this would require a more significant amount of training data and fine-grained shape and text annotation pairs.

6 Conclusion

In order to address the challenge of lacking control in 3D point cloud generation, this paper leverages the inherent structure of 3D shapes. We control the generation of corresponding point cloud shapes for shapes within the same category by inputting different shape structures. We manually annotate the adjacency relationships between segmented parts of each shape, forming the StructureGraph representation. In this graph, nodes represent segmented parts of the point cloud, while edges denote the connectivity relationships between these segmented parts. Utilizing this StructureGraph representation, we propose StrucADT, a novel structure-controllable point cloud generation model built upon the part adjacency conditioned diffusion Transformer model. The StructureGraphNet module in StrucADT extracts structure-aware latent features, whose distributions are then learned by the cCNF module controlled by part adjacency, and both the latent features and part adjacency are incorporated into the Diffusion Transformer module as conditional context to produce structure-controlled point cloud shapes. Experimental results indicate that our structure-controllable 3D point cloud generation method achieves state-of-the-art performance on the ShapeNet dataset, generating high-quality and diverse point cloud shapes while allowing users to control the generation of corresponding point cloud shapes based on the input shape structures.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (62172356, 61872321, 62272277), Zhejiang Provincial Natural Science Foundation of China (LZ25F020012), the Ningbo Major Special Projects of the "Science and Technology Innovation 2025" (2020Z005, 2020Z007, 2021Z012, 2025Z030).

References

[1] Q.-C. Xu, T.-J. Mu, and Y.-L. Yang, "A survey of deep learning-based 3D shape generation," *Computational Visual Media*, vol. 9, no. 3, pp. 407–442, 2023.

[2] Z. Shi, S. Peng, Y. Xu, A. Geiger, Y. Liao, and Y. Shen, "Deep generative models on 3D representations: A survey," *arXiv preprint arXiv:2210.15663*, 2022.

[3] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta, "Learning a predictable and generative vector representation for objects," in *European Conference on Computer Vision*, 2016, pp. 484–499.

[4] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Octree generating networks: Efficient convolutional architectures for high-resolution 3D outputs," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2088–2096.

[5] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling," in *Advances in Neural Information Processing Systems*, 2016, pp. 82–90.

[6] K. Chen, C. B. Choy, M. Savva, A. X. Chang, T. Funkhouser, and S. Savarese, "Text2Shape: Generating shapes from natural language by learning joint embeddings," in *Asian Conference on Computer Vision*, 2019, pp. 100–116.

[7] L. Zhou, Y. Du, and J. Wu, "3D shape generation and completion through point-voxel diffusion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5826–5835.

[8] P. Mittal, Y.-C. Cheng, M. Singh, and S. Tulsiani, "AutoSDF: Shape priors for 3D completion, reconstruction and generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 306–315.

[9] G. Yang, X. Huang, Z. Hao, M.-Y. Liu, S. Belongie, and B. Hariharan, "PointFlow: 3D point cloud generation with continuous normalizing flows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4541–4550.

[10] S. Luo and W. Hu, "Diffusion probabilistic models for 3D point cloud generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2837–2845.

[11] A. Nichol, H. Jun, P. Dhariwal, P. Mishkin, and M. Chen, "Point-E: A system for generating 3D point clouds from complex prompts," *arXiv preprint arXiv:2212.08751*, 2022.

[12] G. K. Nakayama, M. A. Uy, J. Huang, S.-M. Hu, K. Li, and L. Guibas, "DiffFacto: Controllable part-based 3D point cloud generation with cross diffusion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 14 257–14 267.

[13] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "A Papier-Mâché approach to learning 3D surface generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 216–224.

[14] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, "Pixel2Mesh: Generating 3D mesh models from single RGB images," in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 52–67.

[15] C. Wen, Y. Zhang, Z. Li, and Y. Fu, "Pixel2Mesh++: Multi-View 3D mesh generation via deformation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1042–1051.

[16] C. Nash, Y. Ganin, S. A. Eslami, and P. Battaglia, "PolyGen: An autoregressive generative model of 3D meshes," in *International Conference on Machine Learning*, 2020, pp. 7220–7229.

[17] Y. Siddiqui, A. Alliegro, A. Artemov, T. Tommasi, D. Sirigatti, V. Rosov, A. Dai, and M. Nießner, "MeshGPT: Generating triangle meshes with decoder-only Transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 19 615–19 625.

[18] A. Alliegro, Y. Siddiqui, T. Tommasi, and M. Nießner, "PolyDiff: Generating 3D polygonal meshes with diffusion models," *arXiv preprint arXiv:2312.11417*, 2023.

[19] J. Li, K. Xu, S. Chaudhuri, E. Yumer, H. Zhang, and L. Guibas, "GRASS: Generative recursive autoencoders for shape structures," *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 1–14, 2017.

[20] K. Mo, P. Guerrero, L. Yi, H. Su, P. Wonka, N. J. Mitra, and L. J. Guibas, "StructureNet: Hierarchical graph networks for 3D shape generation," *ACM Transactions on Graphics*, vol. 38, no. 6, pp. 1–19, 2019.

[21] L. Gao, J. Yang, T. Wu, Y.-J. Yuan, H. Fu, Y.-K. Lai, and H. Zhang, "SDM-NET: Deep generative network for structured deformable mesh," *ACM Transactions on Graphics*, vol. 38, no. 6, pp. 1–15, 2019.

[22] K. Mo, S. Zhu, A. X. Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. Su, "PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 909–918.

[23] J. Yang, K. Mo, Y.-K. Lai, L. J. Guibas, and L. Gao, "DSG-Net: Learning disentangled structure and geometry for 3D shape generation," *ACM Transactions on Graphics*, vol. 42, no. 1, pp. 1–17, 2022.

[24] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real NVP," in *International Conference on Learning Representations*, 2017, pp. 51–82.

[25] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," in *Advances in Neural Information Processing Systems*, vol. 31, 2018, pp. 10 236–10 245.

[26] D. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *International Conference on Machine Learning*, 2015, pp. 1530–1538.

[27] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in *Advances in Neural Information Processing Systems*, 2018, pp. 6572–6583.

[28] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *International Conference on Machine Learning*, 2015, pp. 2256–2265.

[29] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in *Advances in Neural Information Processing Systems*, 2019, pp. 11 918–11 930.

[30] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems*, 2020, pp. 6840–6851.

[31] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *International Conference on Learning Representations*, 2021, pp. 1735–1756.

[32] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *International Conference on Machine Learning*, 2021, pp. 8162–8171.

[33] P. Dhariwal and A. Nichol, "Diffusion models beat GANs on image synthesis," in *Advances in Neural Information Processing Systems*, 2021, pp. 8780–8794.

[34] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-Based generative modeling through stochastic differential equations," in *International Conference on Learning Representations*, 2021, pp. 3445–3480.

[35] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-Resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 684–10 695.

[36] S. Chaudhuri, D. Ritchie, J. Wu, K. Xu, and H. Zhang, "Learning generative models of 3D structures," *Computer Graphics Forum*, vol. 39, no. 2, pp. 643–666, 2020.

[37] X. Chen, A. Golovinskiy, and T. Funkhouser, "A benchmark for 3D mesh segmentation," *ACM Transactions on Graphics*, vol. 28, no. 3, pp. 1–12, 2009.

[38] Y. Wang, S. Asafi, O. van Kaick, H. Zhang, D. Cohen-Or, and B. Chen, "Active co-analysis of a set of shapes," *ACM Transactions on Graphics*, vol. 31, no. 6, pp. 1–10, 2012.

[39] H. Maron, M. Galun, N. Aigerman, M. Trope, N. Dym, E. Yumer, V. G. Kim, and Y. Lipman, "Convolutional neural networks on surfaces via seamless toric covers," *ACM Transactions on Graphics*, vol. 36, pp. 1–10, 2017.

[40] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su *et al.*, "ShapeNet: An information-rich 3D model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[41] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 6000–6010.

[42] W. Peebles and S. Xie, "Scalable diffusion models with Transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4195–4205.

[43] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3D point clouds," in *International Conference on Machine Learning*, 2018, pp. 40–49.

[44] D. Valsesia, G. Fracastoro, and E. Magli, "Learning localized generative models for 3D point clouds via graph convolution," in *International Conference on Learning Representations*, 2018, pp. 65–79.

[45] D. W. Shu, S. W. Park, and J. Kwon, "3D point cloud generative adversarial network based on tree structured graph convolutions," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3859–3868.

[46] J. Ren, J. Schneider, M. Ovsjanikov, and P. Wonka, "Joint graph layouts for visualizing collections of segmented meshes," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 9, pp. 2546–2558, 2017.

[47] A. Hertz, O. Perel, R. Giryes, O. Sorkine-Hornung, and D. Cohen-Or, "SPAGHETTI: Editing implicit shapes through part aware generation," *ACM Transactions on Graphics*, vol. 41, no. 4, pp. 1–20, 2022.

[48] J. Koo, S. Yoo, M. H. Nguyen, and M. Sung, "SALAD: Part-level latent diffusion for 3D shape generation and manipulation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 14 441–14 451.

[49] B. Zhang, J. Tang, M. Niessner, and P. Wonka, "3DShape2VecSet: A 3D shape representation for neural fields and generative diffusion models," *ACM Transactions on Graphics*, vol. 42, no. 4, pp. 1–16, 2023.

[50] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3D object reconstruction from a single image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 605–613.

[51] E. Sella, G. Fiebelman, N. Atia, and H. Averbuch-Elor, "Spice· E: Structural priors in 3D diffusion using cross-entity attention," in *ACM SIGGRAPH*, 2024, pp. 1–11.

[52] L. Zhang, Z. Wang, Q. Zhang, Q. Qiu, A. Pang, H. Jiang, W. Yang, L. Xu, and J. Yu, "CLAY: A controllable large-scale generative model for creating high-quality 3D assets," *ACM Transactions on Graphics*, vol. 43, no. 4, pp. 1–20, 2024.

[53] W. Cheng and Y. Shan, "Learning layout generation for virtual worlds," *Computational Visual Media*, vol. 10, no. 3, pp. 577–592, 2024.

[54] Y. Li, J. Xiao, Y. Wang, and Z. Lu, "Depthgan: Gan-based depth generation from semantic layouts," *Computational Visual Media*, vol. 10, no. 3, pp. 505–522, 2024.

[55] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*, 2018, pp. 312–323.

[56] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, and L. Guibas, "A scalable active framework for region annotation in 3D shape collections," *ACM Transactions on Graphics*, vol. 35, no. 6, pp. 1–12, 2016.

[57] T. J. McCabe, "A complexity measure," *IEEE Transactions on software Engineering*, no. 4, pp. 308–320, 1976.

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

[59] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," in *Advances in neural information processing systems*, 2015, pp. 3483–3491.

[60] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, 2020, pp. 1877–1901.

Zhenyu Shu earned his PhD degree in 2010 at Zhejiang University, China. He is now working as a full professor at NingboTech University. His research interests include image processing, computer graphics, digital geometry processing, and machine learning. He has published over 40 papers in international conferences or journals.

Jiajun Shen is a graduate student of the School of Software Technology at Zhejiang University. His research interests include computer graphics, geometric processing and computer vision.

Zhonggui Chen received BSc. and Ph.D. degrees in applied mathematics from Zhejiang University, in 2004 and 2009, respectively. He is working as a full professor in the School of Informatics, Xiamen University. His research interests include computer graphics and computational geometry.

Xiaoguang Han is now an Assistant Professor and President Young Scholar of the Chinese University of Hong Kong (Shenzhen) and the Future Intelligence Network Research Institute. He received his PhD degree from the University of Hong Kong in 2017. His research interests include computer vision and computer graphics. He has published nearly 50 papers in well-known international journals and conferences, including top conferences and journals SIGGRAPH (Asia), IEEE TVCG, CVPR, ICCV, ECCV, NeurIPS, ACM TOG, etc. He currently serves as an associate editor of IEEE Transactions on Visualization and Computer Graphics.

Shiqing Xin is a full professor at the School of Computer Science and Technology at Shandong University. He received his PhD degree in applied mathematics at Zhejiang University in 2009. His research interests include image processing, computer graphics, computational geometry, and 3D printing.