Supplementary Materials: Generalized Deep 3D Shape Prior via Part-Discretized Diffusion Process

Yuhan Li[1] Yishun Dou[2] Xuanhong Chen[1] Bingbing Ni[1,2†] Yilin Sun[1] Yutian Liu[1] Fuzhen Wang[1]
[1]Shanghai Jiao Tong University, Shanghai 200240, China [2]Huawei
{melodious, nibingbing}@sjtu.edu.cn
https://github.com/colorful-liyu/3DQD

We first provide the implementation details of the P-VQ-VAE, discrete diffusion generator and condition pipeline in Sec. 1. More ablation study about important settings is reported in Sec. 2. Technical details about experiments are given in Sec. 3, with more visual results in Sec. 4.

1. Implementation

1.1. P-VQ-VAE Backbone

Architecture details. Our P-VQ-VAE backbone consists of three components: an encoder E, a decoder D and a vector quantizer VQ with convolutions. Following AutoSDF [8], we adapt the VQ-VAE from the VAE backbone of LDM [11]. We show the details of encoder in Tab. 1, the decoder in Tab. 3, and the vector quantizer in Tab. 2.

Dataset details. We train the P-VQ-VAE using the objects from 13 categories of ShapeNet [2] data, including [*airplane, bench, cabinet, car, chair, display, lamp, speaker, rifle, sofa, table, phone, watercraft*]. We first extract the Truncated-SDF (T-SDF) following pre-processing steps in DISN [15] and PixelTransformer [13]. The shapes are normalized to lie in an origin-centered cube in $[-1, 1]^3$, while most shape T-SDFs' absolute values are less than 0.5. The signed distance function is evaluated at locations in a uniformly sampled 64^3 grid. Following AutoSDF [8], we use 0.2 as the threshold to further obtain the T-SDFs representations.

Training details. Then the whole 3D shape in the format of T-SDF $X \in \mathbb{R}^{64\times64\times64}$ is divided into 512 partial regions $X' \in \mathbb{R}^{N\times8\times8\times8}$, and $N = 512$ is the number of non-overlap regions, for directly working on whole shape is computationally unaffordable with cubic increase with resolution. Afterward, all regions are vectorized by the encoder as $Z = E(X) \in \mathbb{R}^{N\times n_z}$, while each patch is treated independently and equally.

Next, we obtain a spatial collection of quantized shape tokens $Z_q = \{z_q^0, \ldots, z_q^{N-1}\}$ by a further quantization step as $z_q^i = VQ(z^i) \in \mathbb{R}^{n_z}$, and their corresponding codebook indexes $S = \{s^0, \ldots, s^{N-1}\}$, where $s^i \in \{0, \ldots, K-1\}$. VQ is a spatial-wise quantizer which maps each partial shape z^i into its closest codebook entry in $\mathcal{Z}$. Complete latent space Z and Z_q is produced by gathering z^i and z_q^i for all location i into grids. Finally, decoder D decodes quantized feature Z_q to output the reconstruction X'. The training objective consists of three parts: reconstruction loss, VQ loss and commitment loss:

$$L = \underbrace{\log P(X|Z_q)}_{Reconstruction\ loss} + \underbrace{\|sg\,[Z] - Z_q\|^2}_{VQ\ loss} + \beta \underbrace{\|Z - sg\,[Z_q]\|^2}_{Commitment\ loss}, \tag{1}$$

where $sg[Z]$ means stopping the gradient of Z, so the optimization will only impact the other item. The learning rate is set to be $1e-4$, which halves every 30 epochs. Adam optimizer is used with betas = [0.5, 0.9].

1.2. Discrete Diffusion Generator

Architecture details. We adopt a transformer-based architecture to model the joint probabilistic distribution prior over 3D shapes. The transformer consists of 16 Ordinary Blocks, 3 Multi-frequency Modules (MFM), and a Final Block with the same setting as Ordinary Blocks, as shown in Fig. 1.

We show the details of the Ordinary Block in Fig. 2. The default dimension of latent feature is $C = 256$. All activation function is *GELU2()*. AdaLayerNorm layers are used to synthesize timestep and class-label information and modulate the whole generation process, since timestep t is an important parameter to control the degree of denoising of outputs, while MLP has a boosting bottleneck of $4 \times C$ channels. The MFM doubles the branches of information flowing in Ordinary Blocks with intra- and inter-communication.

†Corresponding author: Bingbing Ni.

E Layer name	Parameters	Input size	Output size
Conv3D	kernel size 3	$Bs \times 1 \times 8 \times 8 \times 8$	$Bs \times 64 \times 8 \times 8 \times 8$
3D ResNet Block	kernel size 3	$Bs \times 64 \times 8 \times 8 \times 8$	$Bs \times 64 \times 8 \times 8 \times 8$
Down-sample	kernel size 3, stride 2, padding 0	$Bs \times 64 \times 8 \times 8 \times 8$	$Bs \times 64 \times 4 \times 4 \times 4$
3D ResNet Block	kernel size 3	$Bs \times 64 \times 4 \times 4 \times 4$	$Bs \times 128 \times 4 \times 4 \times 4$
Down-sample	kernel size 3, stride 2, padding 0	$Bs \times 128 \times 4 \times 4 \times 4$	$Bs \times 128 \times 2 \times 2 \times 2$
3D ResNet Block	kernel size 3	$Bs \times 128 \times 2 \times 2 \times 2$	$Bs \times 128 \times 2 \times 2 \times 2$
Down-sample	kernel size 3, stride 2, padding 0	$Bs \times 128 \times 2 \times 2 \times 2$	$Bs \times 128 \times 1 \times 1 \times 1$
3D ResNet Block	kernel size 3	$Bs \times 128 \times 1 \times 1 \times 1$	$Bs \times 256 \times 1 \times 1 \times 1$
3D ResNet Block	kernel size 3	$Bs \times 256 \times 1 \times 1 \times 1$	$Bs \times 256 \times 1 \times 1 \times 1$
3D Attention Block	Multi-head 1	$Bs \times 256 \times 1 \times 1 \times 1$	$Bs \times 256 \times 1 \times 1 \times 1$
3D ResNet Block	kernel size 3	$Bs \times 256 \times 1 \times 1 \times 1$	$Bs \times 256 \times 1 \times 1 \times 1$
Group Norm	number of groups 32	$Bs \times 256 \times 1 \times 1 \times 1$	$Bs \times 256 \times 1 \times 1 \times 1$
Swish	-	$Bs \times 256 \times 1 \times 1 \times 1$	$Bs \times 256 \times 1 \times 1 \times 1$
Conv3D	kernel size 3	$Bs \times 256 \times 1 \times 1 \times 1$	$Bs \times 256 \times 1 \times 1 \times 1$

Table 1. Architecture of encoder E of P-VQ-VAE. The default stride is set to 1. The default padding is set to 1.

VQ Layer name	Parameters	Input size	Output size
Conv3D	kernel size 3	$Bs \times 256 \times 1 \times 1 \times 1$	$Bs \times 256 \times 1 \times 1 \times 1$
Quantizer	codebook $K \times 256$	$Bs \times 256 \times 1 \times 1 \times 1$	$Bs \times 256 \times 1 \times 1 \times 1$
Conv3D	kernel size 3	$Bs \times 256 \times 1 \times 1 \times 1$	$Bs \times 256 \times 1 \times 1 \times 1$

Table 2. Architecture of vector quantizer VQ of P-VQ-VAE. The default stride is set to 1. The default padding is set to 1.

Training details. After the traning of P-VQ-VAE, each shape is encoded into quantized shape tokens Z_q with their codebook index map $S = \{s^0, \ldots, s^{N-1}\}$, where $s^i \in \{0, \ldots, K-1\}$. The index map is sent to the forward diffusing process followed by reverse learnable denoising chain. In this stage, the encoder and the decoder are frozen, and only the diffusion generator network is trained using the objectives with uniformly sampled timestep t from $\{1, 2, \ldots, 100\}$:

$$L = -\mathbb{E}_{q(s_0)q(s_t|s_0)} \left[\underbrace{\log p_\theta(s_{t-1}|s_t)}_{L_{main}} + \lambda \underbrace{\log p_\theta(\hat{s}_0|s_t)}_{L_{aux}} \right], \tag{2}$$

where λ is $1e-3$.

Following minGPT [10], we separate out all parameters of the model into two buckets: some will experience weight decay for regularization and the others won't (*i.e.* biases, and layernorm / embedding weights). The learning rate is set to be $1e-4$, which multiplies 0.9 every 30 epochs. Besides, to stabilize the training, we convert one-hot row vector s to log one-hot vector with a minimum value of -69.07 ($ln(-30)$) according to Argmax Flow [3].

1.3. Condition Injection Module

In face of the challenge of various conditional generation tasks like shape completion, or generation based on different modalities like images and language, considering a single way to solve all problems is not possible. We design two ways to inject the different modality information which are concisely introduced in main body of this paper. More details are given here.

Shape initialization as conditions. Generally speaking, in the inference stage, the diffusion generator starts the reverse denoising process from completely corrupted or masked shape tokens S_T and recovers the token maps S_T towards the desired data distribution S_0. In this case, the token maps S_T are from the prior setting (*i.e.*, fully masked matrices). However, if the shapes or parts of shapes $\tilde{s}_0$ (*e.g.* the editing task and completion task with given shapes) are provided, we will go on the forward diffusing process for k timesteps on $\tilde{s}_0$ and obtain partially corrupted conditional feature maps $\tilde{s}_k$, where some original shape structure information still remains. Initialized with $\tilde{s}_k$ rather than fully masked tokens S_T, the learned reverse process can be derived as:

$$p_\theta(\tilde{s}_{0:k}) = p(\tilde{s}_k) \textstyle\prod_{t=1}^{k} p_\theta(\tilde{s}_{t-1}|\tilde{s}_t), \tag{3}$$

gradually removing the added noise and generating a 3D shape from condition. Smaller k value means corrupting the condition sightly, so the freedom to denoise and reorganize conditionally initialized token maps is quite restricted,

D Layer name	Parameters	Input size	Output size
Conv3D	kernel size 3	$Bs \times 256 \times 1 \times 1 \times 1$	$Bs \times 256 \times 1 \times 1 \times 1$
3D ResNet Block	kernel size 3	$Bs \times 256 \times 1 \times 1 \times 1$	$Bs \times 256 \times 1 \times 1 \times 1$
3D Attention Block	Multi-head 1	$Bs \times 256 \times 1 \times 1 \times 1$	$Bs \times 256 \times 1 \times 1 \times 1$
3D ResNet Block	kernel size 3	$Bs \times 256 \times 1 \times 1 \times 1$	$Bs \times 256 \times 1 \times 1 \times 1$
Up-sample	kernel size 3, stride 2, padding 0	$Bs \times 256 \times 1 \times 1 \times 1$	$Bs \times 256 \times 2 \times 2 \times 2$
3D ResNet Block	kernel size 3	$Bs \times 256 \times 2 \times 2 \times 2$	$Bs \times 128 \times 2 \times 2 \times 2$
3D Attention Block	Multi-head 1	$Bs \times 128 \times 2 \times 2 \times 2$	$Bs \times 128 \times 2 \times 2 \times 2$
Up-sample	kernel size 3, stride 2, padding 0	$Bs \times 128 \times 2 \times 2 \times 2$	$Bs \times 128 \times 4 \times 4 \times 4$
3D ResNet Block	kernel size 3	$Bs \times 128 \times 4 \times 4 \times 4$	$Bs \times 64 \times 4 \times 4 \times 4$
Up-sample	kernel size 3, stride 2, padding 0	$Bs \times 64 \times 4 \times 4 \times 4$	$Bs \times 64 \times 8 \times 8 \times 8$
3D ResNet Block	kernel size 3	$Bs \times 64 \times 8 \times 8 \times 8$	$Bs \times 64 \times 8 \times 8 \times 8$
3D ResNet Block	kernel size 3	$Bs \times 64 \times 8 \times 8 \times 8$	$Bs \times 64 \times 8 \times 8 \times 8$
Group Norm	number of groups 32	$Bs \times 64 \times 8 \times 8 \times 8$	$Bs \times 64 \times 8 \times 8 \times 8$
Swish	-	$Bs \times 64 \times 8 \times 8 \times 8$	$Bs \times 64 \times 8 \times 8 \times 8$
Conv3D	kernel size 3	$Bs \times 64 \times 8 \times 8 \times 8$	$Bs \times 1 \times 8 \times 8 \times 8$

Table 3. Architecture of decoder D of P-VQ-VAE. The default stride is set to 1. The default padding is set to 1.

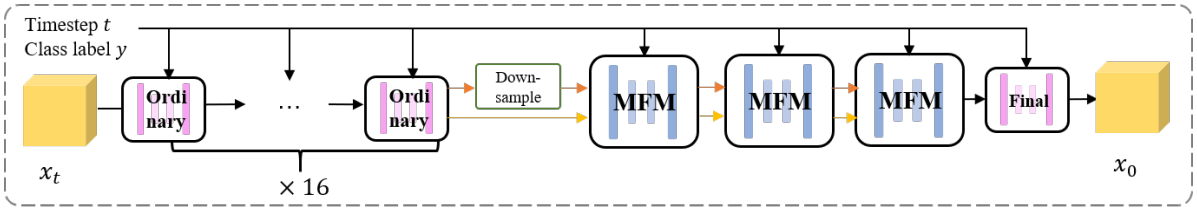

Figure 1. The complete denoising transformer framework which is used in reverse process.

leaning towards less diversity and limited rebuilding capability (*i.e.*, worse TMD and MMD). While a large k value means hurting the condition seriously, it always leads to low-fidelity and distorted generated results, violating the condition (*i.e.*, bad MMD).

We use this kind of condition injection pipeline for shape completion task ($k = 0.5 \times T = 50$), shape denoising task ($k = 0.5 \times T = 50$), and text-driven shape editing ($k = 0.98 \times T = 98$).

Cross-attention for conditions. Although shape initialization solves some problems, not all conditions can be expressed as token maps initialization, especially condition c with different modalities (*e.g.*, images, texts). As a consequence, inspired by [11], we use cross-attention modules to inject conditions. At first, conditions with different modalities are encoded into discrete and quantized tokens embedding with its own encoder. For texts, we utilize pre-trained CLIP [9] model (ViT-B) to project text prompts to 77 tokens c_q. As for single-view images, VQ-VAE from [8] is used to encode images to 512 tokens c_q, which is specifically trained on image data on Pix3D dataset [12] and aligned with shape tokens. Next, only the indexes c_I of these tokens among codebook entries are preserved, while c_q is discarded. The wide use of indexes is beneficial to allow the model to receive uniform input (*i.e.*, conditions and shapes are all expressed with integers), and eliminate the inherent differences in data distribution among different modalities. Then the conditional token indexes are projected to tensors with learnable embedding modules, followed by fully connected layers to encode them to key and $value$ into cross-attention. The Ordinary Block and MFM are modified with a LayerNorm and a cross-attention module behind self-attention, as shown in Fig. 3.

2. More Ablation Study

2.1. Shape novelty analysis.

To statistically analyze the novelty of the generated shapes, we visualize the distribution of LFDs between our generated shapes and retrieved shapes in Fig. 4. Note that our 3DQD can not only learn the distribution of training data (low LFD), but also generate novel shapes (high LFD) that are different from the training data.

2.2. Multi-frequency Fusion Module Settings.

In the main body of this paper, we have analyzed the frequency components of the generation to validate our design

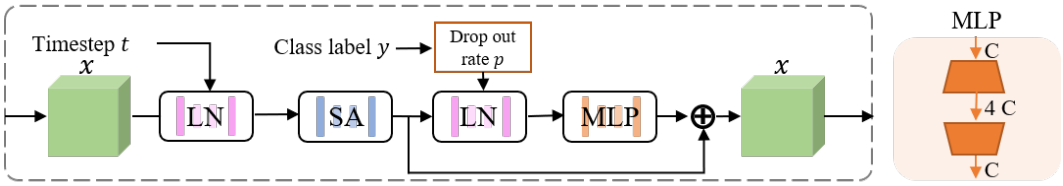

Figure 2. Ordinary block with a boosting bottleneck MLP.

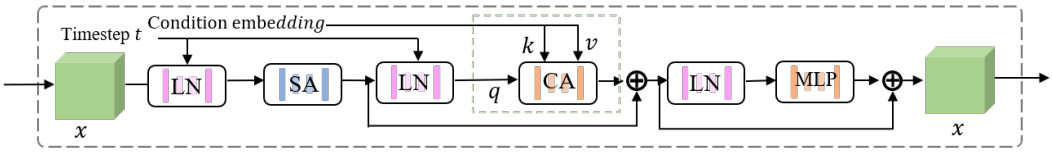

Figure 3. Ordinary block with cross-attention to modulate multi-modality information.

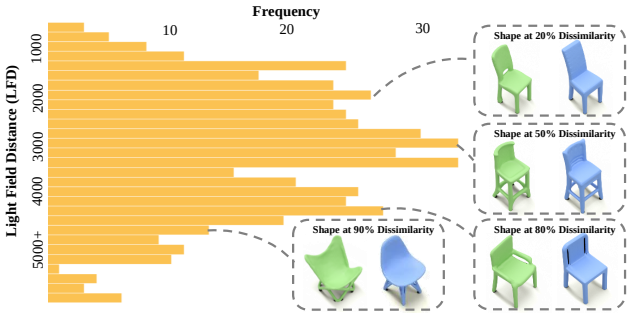

Figure 4. We generate 500 random chairs using 3DQD, then retrieve the most similar shape in training set for each samples with LFD metric. The distribution of LFDs between samples (in green) and retrieved shapes (in blue) is plotted here.

of Multi-frequency Fusion Module (MFM) with quantitative and visual comparisons. In this section, to verify our designed MFM settings, we compare 4 configurations about MFM on the shape completion task:

- 3DQD model with different MFM layers. It still utilizes residual add as Pair-wise Fusion Operator (PFO) with the complete MFM pipeline.
- 3DQD model with cross-attention to fuse two branches of frequency components. In this case, adaptive rescaling is no longer required, for cross-attention is able to align features with different spatial dimensions with QKV. It implements 3 complete MFM layers.
- 3DQD model with single-fusion MFM layers as shown in Fig. 5. In this case, the inter-communication between y^L and x^L is absent, so the fusion only occurs in the other side. It uses residual add as fusion operator with 3 MFM layers.
- 3DQD model with single-attention MFM layers as shown in Fig. 5. In this case, the intra-communication within y^L is absent. It uses residual add as fusion operator with 3 MFM layers.

MFM layers	Fusion types	Dual fusion	Dual-attn	MMD↓
3	residual add	✔	✔	**0.2934**
0				0.3205
1				0.3083
2				0.3001
4				0.2942
5				0.2937
	cross-attn			0.4173
		✘		0.2981
			✘	0.3059

Table 4. Comparing our full MFM pipeline (the first row) with various ablated cases on the shape completion task. We use the hyperparameters on the first line as default hyperparameters for blanks. MMD is multiplied by 1×10^2.

Quantitative results are shown in Tab. 4, where we compare our full MFM pipeline with various ablated cases. The best choice for MFM is utilizing residual add as PFO, with 3 full MFM pipeline layers.

3. Experiment Detail

3.1. Details for Unconditional Generation

Evaluation metrics details. Although the outputs of our model are T-SDFs, we choose to evaluate them by sampling point clouds on their surfaces. The community has proposed different metrics to quantitatively evaluate the generation performance of point cloud generative models, but some of them suffer from certain drawbacks. Given a generated set of point clouds S_g and a reference set S_r, the most popular metrics are (following Lion [18]):

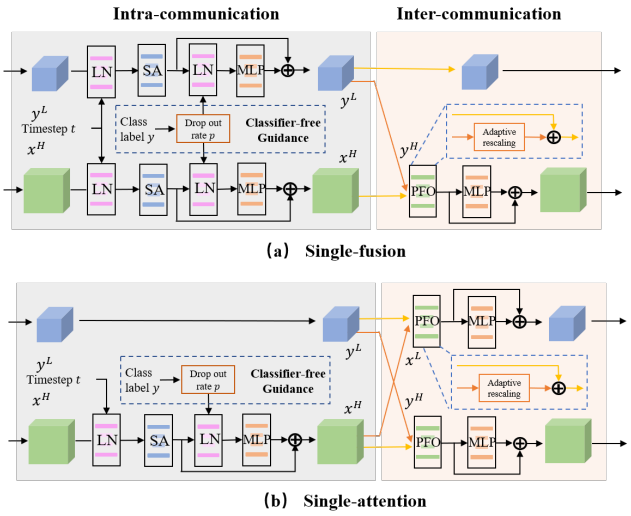

Figure 5. Two MFM communication frameworks in ablation cases (single-fusion and single-attention).

- **Coverage(COV)**:

$$COV(S_g, S_r) = \frac{\left|\left\{\arg\min_{Y \in S_r} D(X,Y) | X \in S_g\right\}\right|}{|S_r|}, \tag{4}$$

where $D(\cdot,\cdot)$ is distance measurement (*i.e.* Chamfer distance or earth mover distance). COV measures the number of reference point clouds that are matched to at least one generated shape. COV can quantify diversity, but low-quality and diverse generated point clouds can achieve high coverage scores.

- **Minimum matching distance(MMD)**:

$$MMD(S_g, S_r) = \frac{1}{|S_r|} \sum_{Y \in S_r} \min_{X \in S_g} D(X,Y). \tag{5}$$

MMD calculates the average distance between the point clouds in the reference set and their closest neighbors in the generated set. However, MMD is not sensitive to low-quality point clouds which will never be matched during calculation. Moreover, MMD varies depending on implementation and is not robust on large scale unconditional shape generation.

- **1-nearest neighbor accuracy (1-NNA)**: To overcome the drawbacks of COV and MMD, PointFlow [16] proposes to use 1-NNA as a metric to evaluate point cloud generative models:

$$1\text{-}NNA(S_g, S_r) = \frac{\sum_{X \in S_g} \mathbb{I}\left[N_X \in S_g\right] + \sum_{Y \in S_r} \mathbb{I}\left[N_Y \in S_r\right]}{|S_r| + |S_g|}, \tag{6}$$

where $\mathbb{I}[\cdot]$ is the indicator function and N_X is the nearest neighbor of X in the set $S_r \cup S_g - \{X\}$. Hence, 1-NNA represents the leave-one-out accuracy of the 1-NN classifier defined in Eq. (6). More specifically, this 1-NN classifier classifies each sample as belonging to either S_r or S_g based on its nearest neighbor sample within N_X. As a consequence, 1-NNA directly quantifies distribution similarity between S_r and S_g and measures both quality and diversity.

In conclusion, we are following PointFlow [16], PVD [20] and use 1-NNA as our main evaluation metric to quantify shape generation performance. with both CD and EMD distances.

More quantitative results. More quantitative results including COV and MMD are reported in Tab. 5 for an additional comparison, although they may be unreliable metrics for unconditional generation quality. 3DQD outperforms all baselines in experiment for the more reasonable 1-NNA metric.

About normalization method. Almost all of the baselines in Tab. 5 preprocess meshes into point clouds following PointFlow, while we preprocess them into T-SDF following DISN [15], which causes **different scales of shapes** in two kinds of datasets. Moreover, the most popular method (PointFlow [16], PVD [20], Lion [18], *etc.*) count the mean and variance of the shapes in the training set, and apply **dataset-wise normalization** to keep the same scales with the training set during evaluation. For a fair comparison on MMD, dataset-wise normalization is also performed on our generated shapes and our test sets to keep the same scales with the training set of PointFlow [16] and PVD [20]. There is still a gap on MMD to some extent, which mainly comes from the loss caused by multiple sampling during data preprocessing and evaluation. So we strongly recommend referring to relative distance metrics, such as 1-NNA metric.

Improvement on airplane. The 20% performance improvement mainly comes from a large number of special shapes in *airplane* category, such as flat fighters and planes with complex propellers. Our 3DQD generates shapes with exact and clean details in Fig. 6, because of sufficient and

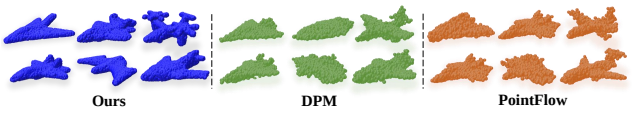

Figure 6. We sample shapes with 3DQD model, then we select samples generated by baselines with minimum CD from ours for a fair comparison. Our method outperforms baselines significantly in special shape cases in *Airplane* due to exact shape tokens.

accurate local shape tokens stored in codebook, while previous works are highly possible to produce failure cases with low-quality shape details.

3.2. Details for Shape Completion

Evaluation setting details. For PoinTr [17] and SeedFormer [19], we use the pre-trained model released on ShapeNet-55 benchmark, which fully covers the 13 categories we used to evaluate. Following PoinTr [17], for each object, we randomly sample 8,192 points from the surface to obtain the ground truth point cloud. During evaluation, the points clouds in partial regions (*i.e.*, half and octant) are preserved, while others are discarded. Then the amount of points preserved is repeated or deleted to 4096 (moderate difficulty degree in their benchmark [17]) as inputs. The models' outputs are 8192 complete points. As for AutoSDF [8] and 3DQD, the partial T-SDFs regions are inputs, and 8192 points are sampled on surfaces of generated shapes. Before computing metrics, all point clouds are normalized with method in SeedFormer [19].

Evaluation metrics details. *Unidirectional Hausdorff Distance* (UHD) is used to measure the completion fidelity to the input partial input. MPC [14] and AutoSDF [8] calculate the average Hausdorff distance from the input partial shape sets S_p to each of the k completion results in S_g:

$$UHD(S_p, S_g) = \frac{1}{|S_p|} \sum_{Z \in S_p, X \in S_g} \left(\frac{1}{k} \sum_{i=1}^{k} D^{HD}(X^i, Z) \right). \tag{7}$$

However, UHD only measures the distortion of generated shapes from partial inputs, rather than the quality of completion with ground truths. Low-quality shapes with inputs unchanged will achieve good scores. As a result, we use MMD and Average Matching Distance (AMD) to evaluate completion quality. MMD and AMD for diverse shape completion can be derived as:

$$MMD(S_g, S_r) = \frac{1}{|S_r|} \sum_{Y \in S_r, X \in S_g} \left\{ \min_{X_i \in X} D(X^i, Y) \right\}.$$
$$AMD(S_g, S_r) = \frac{1}{|S_r|} \sum_{Y \in S_r, X \in S_g} \left\{ \frac{1}{|X|} \sum_{i=0}^{|X|} D(X^i, Y) \right\}. \tag{8}$$

3.3. Details for Language-guided Generation

Training and inference details are reported in Sec. 1.3. As for metrics, *Pairwise Minimum Matching Distance* (PMMD) has the same expression as MMD in Eq. (8). *Frechet Pointcloud Distance* (FPD) is first proposed in TreeGAN to calculate the distance between features of point clouds with a pre-trained PointNet backbone. We select the shapes with minimum PMMD from ground truths in each k generation to build a group of samples, and then measure FPD between the generated data distribution and ground truth distribution on test set, to check if the model can simulate the real shape distribution. Before computing metrics, all point clouds are normalized with method in SeedFormer [19].

3.4. Details Denoising Conditional Generation

The pre-trained single-class 3DQD model trained on *chair* category of ShapeNet data [2] is evaluated about its denoising capability in this section to validate the prior on extended applications. We first sample 500 pure T-SDFs inputs $X \in \mathbb{R}^{64\times64\times64}$ from the dataset. Then different types of noise $\epsilon \in \mathbb{R}^{64\times64\times64}$ (*i.e.* standard Gaussian distribution and uniform distribution) with same spatial resolutions scaled by noise level α are added to pure inputs:

$$X^{noisy} = X + \alpha\epsilon. \tag{9}$$

Afterward, the noisy input X^{noisy} is encoded by P-VQ-VAE and set as shape token initialization $\tilde{s}_k$ following the same procedure in *Shape initialization as conditions* part. $k = 0.5 \times T = 50$. At last reverse process starts from $\tilde{s}_k$ to recover the clean samples and remove the noise ϵ.

Quantitative results with different noise levels and types are reported in Tab. 6, and visual results are in Sec. 4.

3.5. Details for Single-view Reconstruction

Dataset details. We evaluate our proposed method on the real-world benchmark Pix3D [12], using cropped and segmented images as inputs for reconstruction (*i.e.*, the background is removed). Since official train/test splits are only provided for the chair category, we test single-view reconstruction on chair category.

Category	Model	MMD↓		COV↑ (%)		1-NNA↓ (%)	
		CD	EMD	CD	EMD	CD	EMD
Airplane	r-GAN [1]	0.447	2.309	30.12	14.32	99.84	96.79
	l-GAN (CD) [1]	0.340	0.583	38.52	21.23	87.30	93.95
	l-GAN (EMD) [1]	0.397	0.417	38.27	38.52	89.49	76.91
	PointFlow [16]	0.224	0.390	47.90	46.41	75.68	70.74
	SoftFlow [4]	0.231	0.375	46.91	47.90	76.05	65.80
	SetVAE [5]	0.200	0.367	43.70	48.40	76.54	67.65
	DPF-Net [6]	0.264	0.409	46.17	48.89	75.18	65.55
	DPM [7]	0.213	0.572	48.64	33.83	76.42	86.91
	PVD [20]	0.224	0.370	**48.88**	**52.09**	73.82	64.81
	3DQD *(ours)*	0.551	0.426	40.50	47.17	**56.29**	**54.78**
Chair	r-GAN [1]	5.151	8.312	24.27	15.13	83.69	99.70
	l-GAN (CD) [1]	2.589	2.007	41.99	29.31	68.58	83.84
	l-GAN (EMD) [1]	2.811	1.619	38.07	44.86	71.90	64.65
	PointFlow [16]	2.409	1.595	42.90	50.00	62.84	60.57
	SoftFlow [4]	2.528	1.682	41.39	47.43	59.21	60.05
	SetVAE [5]	2.545	1.585	46.83	44.26	55.84	60.57
	DPF-Net [6]	2.536	1.632	44.71	48.79	62.00	58.53
	DPM [7]	2.399	2.066	44.86	35.50	60.05	74.77
	PVD [20]	2.622	1.556	**49.84**	**50.60**	56.26	53.32
	3DQD *(ours)*	2.057	1.128	46.76	48.13	**55.61**	**52.94**
Car	r-GAN [1]	1.446	2.133	19.03	6.539	94.46	99.01
	l-GAN (CD) [1]	1.532	1.226	38.92	23.58	66.49	88.78
	l-GAN (EMD) [1]	1.408	0.899	37.78	45.17	71.16	66.19
	PointFlow [16]	0.901	0.807	46.88	50.00	58.10	56.25
	SoftFlow [4]	1.187	0.859	42.90	44.60	64.77	60.09
	SetVAE [5]	0.882	0.733	49.15	46.59	59.94	59.94
	DPF-Net [6]	1.129	0.853	45.74	49.43	62.35	54.48
	DPM [7]	0.902	1.140	44.03	34.94	68.89	79.97
	PVD [20]	1.077	0.794	41.19	50.56	**54.55**	53.83
	3DQD *(ours)*	0.677	0.443	**49.28**	**55.43**	55.75	**52.80**

Table 5. Generation performance metrics on Airplane, Chair, Car. MMD-CD is multiplied with 1×10^3, MMD-EMD is multiplied with 1×10^2.

Noise type	Level	MMD↓	AMD↓	TMD↑
Gaussian	0.01	0.710	0.897	0.331
	0.02	0.737	0.918	0.343
	0.05	1.006	1.369	0.543
	0.1	3.680	5.043	1.724
Uniform	0.01	0.720	0.906	0.337
	0.02	0.756	0.964	0.355
	0.05	1.086	1.448	0.449
	0.1	2.385	3.167	0.772

Table 6. Shape denoising performance with different types and levels of noise. MMD and AMD is multiplied with 1×10^3, TMD is multiplied with 1×10^2.

Architecture details. AutoSDF [8] has released a VQ-VAE to encode images to 512 tokens c_q, which is aligned with shape tokens. Based on it, we finish single-view reconstruction with pre-trained 3DQD model trained on language-guided generation task as the backbone, for it has cross-attention modules to fuse cross-modality information. Almost all parts of pre-trained model are preserved, except the conditional embedding modules.

Training and inference details. During reconstruction, we use two ways of condition injection (*i.e.*, shape token initialization and cross-attention). Namely, we first encode each image into 512 token indexes s_0 with VQ-VAE. Then the token indexes s_0 after k timestep corruption are initialized as the start of reverse denoising chain $\tilde{s}_k$, where $k = 0.5 \times T = 50$. Afterward, the image indexes s_0 are

also injected into 3DQD with new learnable embedding layers and cross-attention modules, which is the same as text-driven shape generation. Note that this image-condition 3DQD model is fine-tuned on Pix3D [12] with masked and cropped image-shape pairs for 10 hours, to train the new image embedding and refine the whole model well.

4. Additional Experimental Results

More visual results are presented in this section, including unconditional shape generation in Fig. 7, shape completion comparison in Fig. 8 and Fig. 9, denoising results in Fig. 10, single-view reconstruction in Fig. 11.

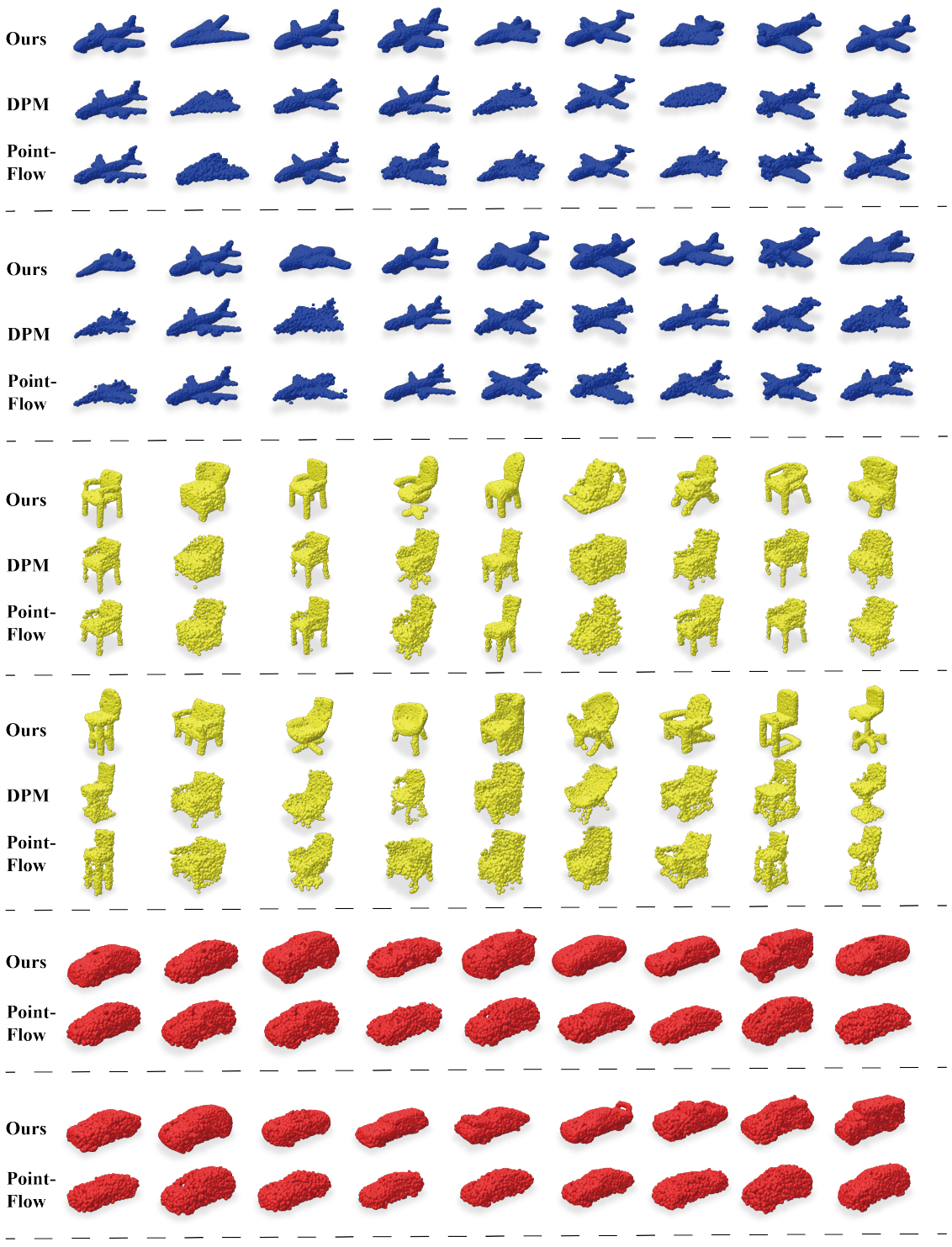

Figure 7. Visual comparison about unconditional shape generation. We randomly sample shapes with our 3DQD model, then we select samples generated by baselines with minimum Chamfer distances from ours, to present a fair comparison.

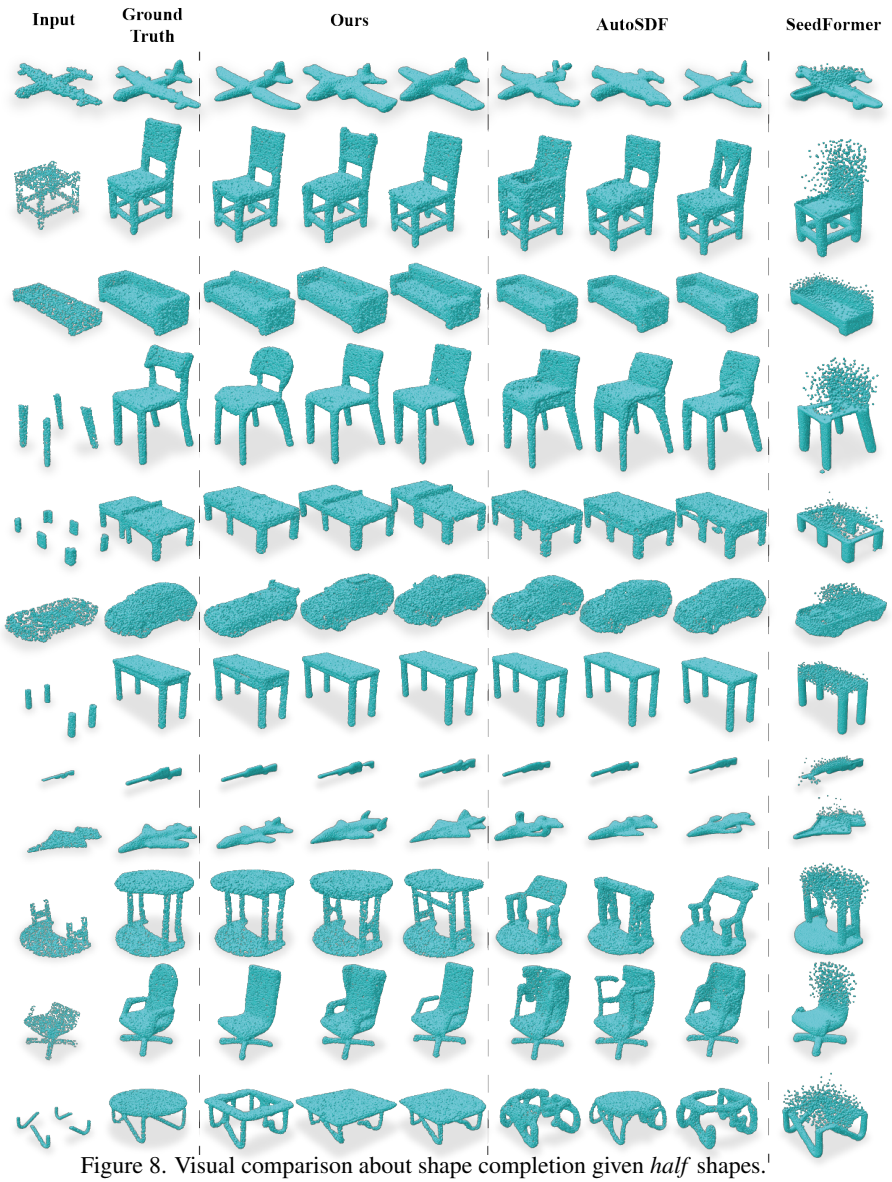

Figure 8. Visual comparison about shape completion given *half* shapes.

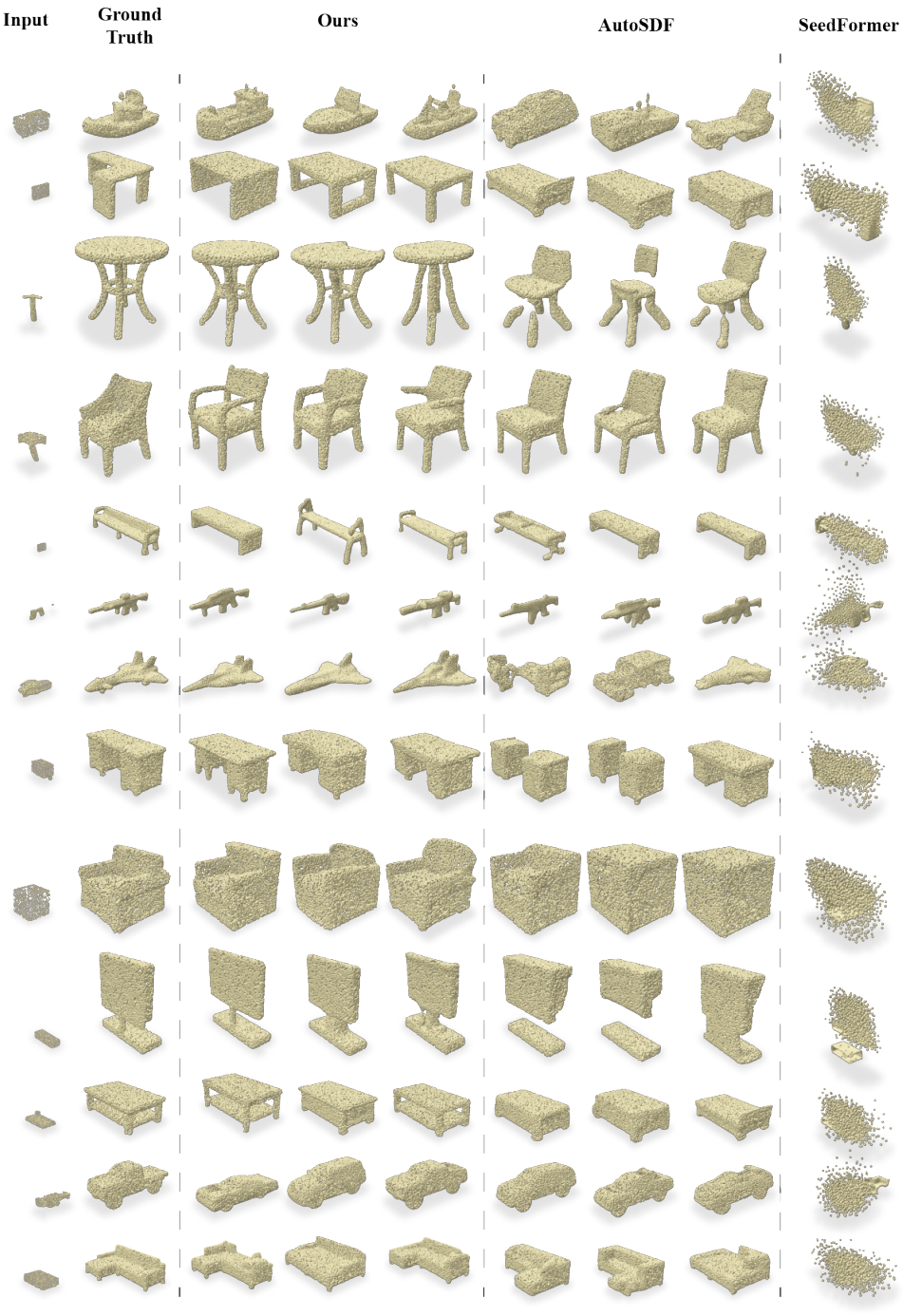

Figure 9. Visual comparison about shape completion given *octant* shapes.

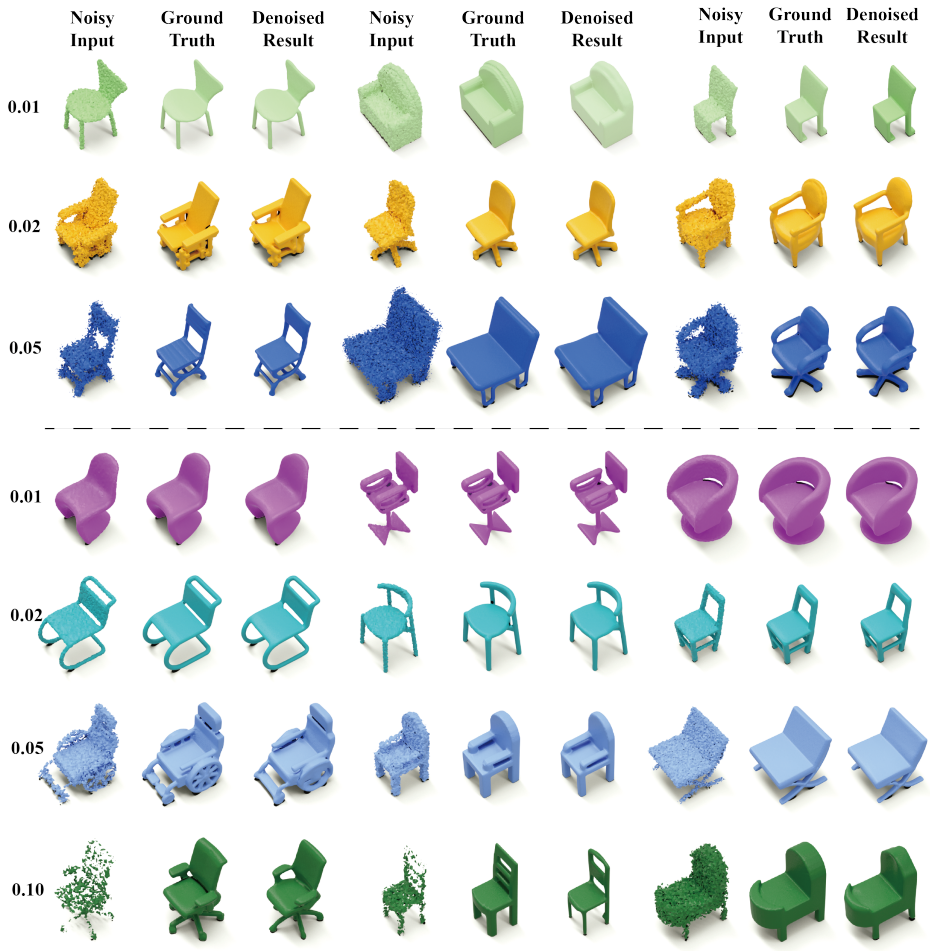

Figure 10. Visual results about denoising with various noise types and levels. *Upper*: Gaussian noise. *Lower*: Uniform noise.

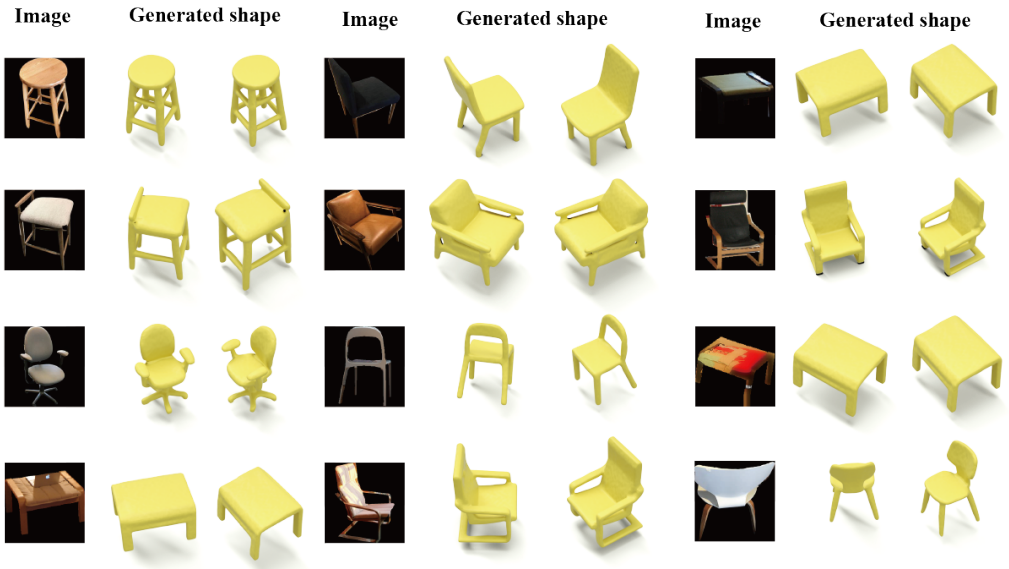

Figure 11. Visual results about single-view reconstruction.

References

[1] 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. 7

[2] 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. 1, 6

[3] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in Neural Information Processing Systems*, 34:12454–12465, 2021. 2

[4] Hyeongju Kim, Hyeonseung Lee, Woo Hyun Kang, Joun Yeop Lee, and Nam Soo Kim. Softflow: Probabilistic framework for normalizing flow on manifolds. *Advances in Neural Information Processing Systems*, 33:16388–16397, 2020. 7

[5] Jinwoo Kim, Jaehoon Yoo, Juho Lee, and Seunghoon Hong. Setvae: Learning hierarchical composition for generative modeling of set-structured data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15059–15068, 2021. 7

[6] 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. 7

[7] 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. 7

[8] Paritosh Mittal, Yen-Chi Cheng, Maneesh Singh, and Shubham Tulsiani. Autosdf: Shape priors for 3d completion, reconstruction and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 306–315, 2022. 1, 3, 6, 7

[9] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 3

[10] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 2

[11] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021. 1, 3

[12] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2974–2983, 2018. 3, 6, 8

[13] Shubham Tulsiani and Abhinav Gupta. Pixeltransformer: Sample conditioned signal generation. *arXiv preprint arXiv:2103.15813*, 2021. 1

[14] Rundi Wu, Xuelin Chen, Yixin Zhuang, and Baoquan Chen. Multimodal shape completion via conditional generative adversarial networks. In *European Conference on Computer Vision*, pages 281–296. Springer, 2020. 6

[15] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. *Advances in Neural Information Processing Systems*, 32, 2019. 1, 5

[16] 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. 5, 7

[17] Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12498–12507, 2021. 6

[18] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. *arXiv preprint arXiv:2210.06978*, 2022. 4, 5

[19] Haoran Zhou, Yun Cao, Wenqing Chu, Junwei Zhu, Tong Lu, Ying Tai, and Chengjie Wang. Seedformer: Patch seeds based point cloud completion with upsample transformer. *arXiv preprint arXiv:2207.10315*, 2022. 6

[20] 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. 5, 7