Learning Versatile 3D Shape Generation with Improved Auto-regressive Models

Simian Luo[1,*], Xuelin Qian[1,*], Yanwei Fu[1,†], Yinda Zhang[2], Ying Tai[3]
Zhenyu Zhang[3], Chengjie Wang[3], Xiangyang Xue[1]
[1]Fudan University; [2]Google; [3]Tencent Youtu Lab
{18300180157,xlqian,yanweifu,xyxue}@fudan.edu.cn, yindaz@gmail.com,
zhangjesse@foxmail.com, {yingtai,jasoncjwang}@tencent.com

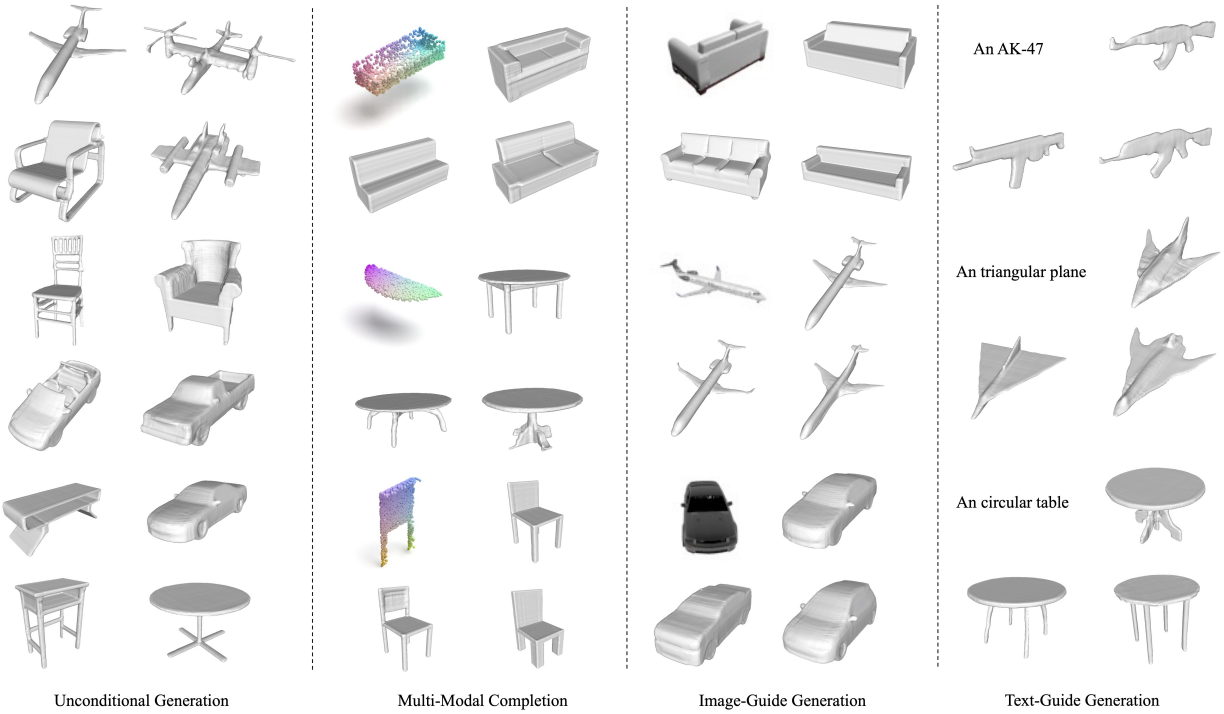

Figure 1: We propose an improved AR model (ImAM) to learn versatile 3D shape generation. ImAM can either generate diverse and faithful shapes with multiple categories via an unconditional way (one column to the left), or can be adapted for conditional generation by incorporating various conditioning inputs given on the left-top (three columns to the right).

Abstract

Auto-Regressive (AR) models have achieved impressive results in 2D image generation by modeling joint distributions in the grid space. While this approach has been extended to the 3D domain for powerful shape generation, it still has two limitations: expensive computations on volumetric grids and ambiguous auto-regressive order along grid dimensions. To overcome these limitations, we propose the Improved Auto-regressive Model (ImAM) for 3D shape generation, which applies discrete representation learning based on a latent vector instead of volumetric grids. Our approach not only reduces computational costs but also preserves essential geometric details by learning the joint distribution in a more tractable order. Moreover, thanks to the simplicity of our model architecture, we can naturally extend it from unconditional to conditional generation by concatenating various conditioning inputs, such as point clouds, categories, images, and texts. Extensive experiments demonstrate that ImAM can synthesize diverse and faithful shapes of multiple categories, achieving state-of-the-art performance.

[*]Equal contribution
[†]Corresponding author

1. Introduction

3D shape generation has garnered increasing interest in both academia and industry for its extensive applications in robotics [20], autonomous driving [47, 28], augmented reality [35] and virtual reality [34]. Based on whether user prerequisites are provided, shape generation is typically categorized as unconditional or conditional. To be an effective generative model, it is crucial for the synthesized shapes to be both *diverse* and *faithful* to the universal cognition of humans or given conditions. These qualities serve as the foundation for other deterministic tasks, such as shape completion, single-view reconstruction, and more. Previous approaches [8, 14, 25] usually utilize an AutoEncoder (AE) to learn latent features by shape reconstruction. Then, a GAN is trained to fit the distributions of the latent features, allowing for the generation of 3D shapes through sampling the latent codes learned in AE. While achieving convincing results, a single embedding for one shape easily encounters the problem of poor scalability and difficulty in training.

Recently, Auto-Regressive (AR) models have shown remarkable performance in the generation of 2D images [11, 50, 5] and 3D shape [23, 45]. Instead of learning a continuous latent space, these model leverage discrete representation learning to encode each 2D/3D input into grid-based discrete codes. Subsequently, a transformer-based network is employed to jointly model the distribution of all codes, which essentially reflects the underlying prior of objects, facilitating high-quality generation and tractable training. However, applying AR models to 3D still suffers from two limitations. First, as the number of discrete codes increases from squared to cubed, the computational burden of the transformer grows dramatically, making it difficult to converge. Second, discrete codes in the grid space are highly coupled. It is ambiguous to simply flatten them for auto-regression (*e.g.*, a top-down row-major order). This may lead to poor quality or even collapse of generated shapes (see Sec. 3.1 and the Supplementary for more details).

In this paper, we propose an improved auto-regressive model (ImAM), to enhance the efficient learning of 3D shape generation. Our key idea is to apply discrete representation learning in a one-dimensional space instead of 3D volumetric space. Specifically, we first project volumetric grids encoded from 3D shapes onto three axis-aligned orthogonal planes. This process significantly reduces the computational costs from cubed to squared while maintaining the essential information about the input geometry. Next, we present a coupling network to further encode three planes into a compact and tractable latent space, on which discrete representation learning is performed.

Our ImAM is straightforward and effective, simply tackling the aforementioned limitations by two projections. Thus, a vanilla decoder-only transformer can be attached to model the joint distributions of codes from the latent spaces. Furthermore, the simplicity of the transformer structure allows us to switch freely between unconditional and conditional generation by concatenating various conditioning inputs, such as point clouds, categories, images and texts. Figure 1 showcases the ability of our ImAM to generate diverse and accurate shapes across multiple categories, both with and without the given condition on the top-left corner.

In summary, the contributions of this paper are listed as follows. (1) We propose an improved AR Model (ImAM) for 3D shape generation. By applying discrete representation learning in a latent vector instead of volumetric grids, our ImAM enjoys the advantages of lightweight and flexibility. (2) Our proposed ImAM model provides a more unified framework for switching between unconditional and conditional generation for a variety of conditioning inputs, including point clouds, categories, images, and texts. (3) Extensive experiments are conducted on four tasks to demonstrate that our ImAM can generate more faithful and diverse shapes, achieving state-of-the-art results for unconditional and conditional shape generation. Overall, our contributions advance the field of 3D shape generation, providing a powerful tool for researchers and practitioners alike.

2. Related work

3D shape generative models. As an extremely challenging task, we review the most previous efforts by using voxel, point clouds, and implicit representations. (1) Standard voxel grids can be easily processed by 3D convolution for learning-based 3D task [19, 44, 10, 41]. However, restricted by its cubic space complexity, voxel representation can not scale to a high resolution, usually limited in 64^3. Even with efficient data structures like octrees or multi-scale representation [40, 13, 30], such representations still have some limitations in quality. (2) Point clouds extracted from shape surfaces is an alternative 3D representation [12, 43, 27], which is efficient in terms of memory and does not suffer from the restriction of resolution compared with voxel representations. However, these representation can not represent topological relations in 3D space and are also nontrivial to recover shape surfaces from point cloud representation. (3) Recently, implicit representations have gained attention for simplicity in representing 3D shapes [25, 21, 22, 33]. By predicting the signed distance [25, 33] or occupancy label [21, 26] of a given point, and then through Marching cubes [18] methods, the surface can be easily recovered. Follow-up works [21, 25, 8, 26, 15] focus on the design of implicit function representation with global or local shape priors. To sum up, different 3D representations lead to various shape generative models. There are various good previous works such as 3DGAN [41], PC-GAN [1], IM-GAN [8], and GBIF [14]. However, most current generative methods are task-specific. And it is difficult to be directly applied to different generative tasks (*e.g.*, shape completion). Fundamentally,

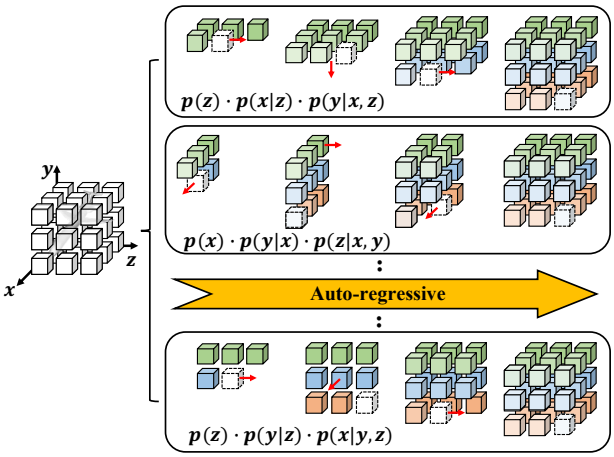

Figure 2: Illustration of auto-regressive generation for grid-based representation. Here, we show three different flattening orders as examples. Best viewed in color.

they rely on GANs for the generation step, suffering from the known drawbacks such as mode collapse and training instability. In contrast, we propose an improved AR model for 3D shape generation that can synthesize high-quality and diverse shapes while being easily generalized to other multi-modal conditions.

Autoregressive models. AR models are probabilistic generative approaches that have tractable probability density. Using the probability chain rule, the likelihood of a given sample (usually high dimensional data) can be factorized into a series of product of conditional probability. In contrast, GANs do not have such a tractable probability density. Recently, AR models achieve remarkable progress in 2D image generation [11, 37, 29], to a less extent 3D tasks [36, 9, 24]. Most of 3D AR models are struggling with generating high-quality shapes due to the challenges of representation learning with more points or faces. Particularly, we notice two recent works [45, 23] that share similar insights of utilizing AR models for 3D tasks. [45] introduces a sparse representation to only quantize non-empty grids in 3D space, but still follows a monotonic row-major order. [23] presents a non-sequential design to break the orders, but performs on all volumetric grids. However, both of them address only one of the above limitations, and burden the structure design and training of transformer. In contrast, our ImAM applies discrete representation learning in a latent vector instead of volumetric grids. Such a representation offers plenty of advantages, including shorter length of discrete codes, tractable orders from auto-regression, fast convergence, and also preserving essential 3D information. Moreover, benefiting from the simplicity of transformers, we can freely switch from unconditional generation to conditional generation by concatenating various conditions.

3. Methodology

Figure 3 illustrates the schematic of our proposed framework for 3D shape generation, which consists of a two-stage training procedure. It first represents the input as a composition of learned discrete codes (in Sec. 3.2), then utilizes a transformer model to learn their interrelations (in Sec. 3.3). Before beginning, we provide some symbol definitions and necessary preliminaries in Sec. 3.1.

3.1. Preliminary

Ambiguity. Formally, *'ambiguity' appears in the order of a series of conditional probabilities, which affects the difficulty of likelihood learning, leading to approximation error of the joint distribution.* Critically, auto-regressive models requires sequential outputs, autoregressively predicting the next code conditioned on all previous ones. Thereby, the order of the flattened sequence determines the order of conditional probabilities. Although some methods (*e.g.* position embedding [39]) can be aware of positions of codes, it cannot eliminate approximation error caused by the condition order. Notably, this 'ambiguity' phenomenon is also discussed in [11] (in Fig. 47), where the loss curves in Fig. 47 highlight the differences in difficulty for likelihood learning across various orders. Figure 2 illustrates how the flattening order affects the way of autoregressive generation. For grid-based representation, it is ambiguous if the flattening order along axes is y-x-z, x-z-y or other combinations.

Discrete Representation. Given input point clouds $P \in \mathbb{R}^{n\times 3}$ where n means the number of points, an encoder is adopted to extract features for each point cloud and then perform voxelization to get features of regular volumetric grids $f^v \in \mathbb{R}^{r\times r\times r\times c}$, where r denote the resolution of voxels and c is the feature dimension. To learn discrete representation for each 3D shape, a codebook $\mathbf{q} \in \mathbb{R}^{m\times c}$ is thus introduced whose entry is a learned code describing a particular type of local-part shape in a grid. Formally, for each grid $\left\{f^v_{(h,l,w)}\right\}^r_{h,l,w=1}$, vector quantization $\mathcal{Q}(\cdot)$ is performed by replacing it with the closest entry in codebooks [11],

$$\mathbf{z}^v = \mathcal{Q}(f^v) := \arg\min_{\mathbf{e}_i\in\mathbf{q}} ||f^v_{(h,l,w)} - \mathbf{e}_i|| \qquad (1)$$

where $\mathbf{e}_i \in \mathbf{q}$ represents the i-th entry in the codebook. Thus, learning the correlations between entries in the second stage can explore the underlying priors for shape generation. However, autoregressive generation [11, 45] requires sequential outputs, facing two limitations. *First, the resolution of $\mathbf{z}^v$ matters the quality of synthesized shapes*. If r is too small, it lacks the capacity to represent intricate and detailed geometries, while a large value of r can learn a specific code for each local gird, it inevitably increase the computational complexity since the number of required codes

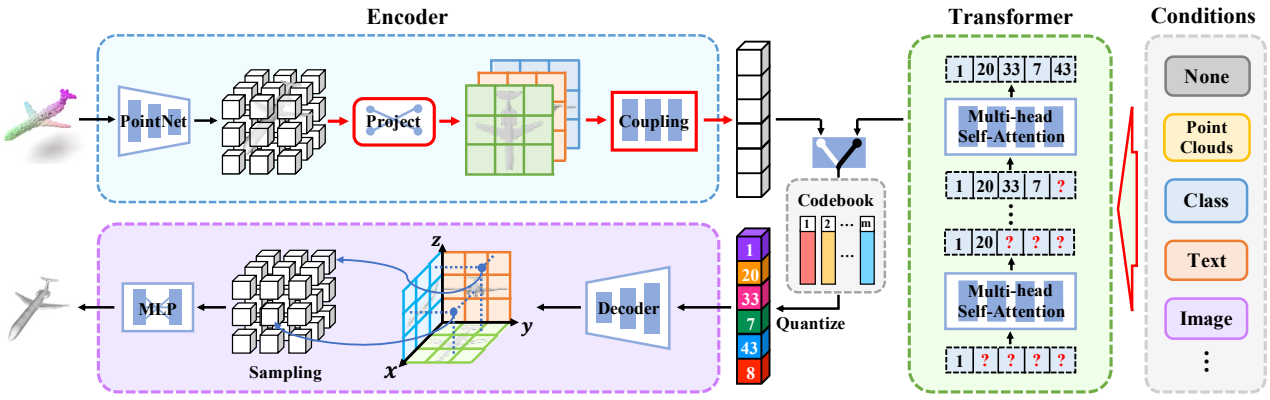

Figure 3: Overview of our ImAM. Given an arbitrary 3D shape, we first project encoded volumetric grids into the three axis-aligned planes, and then use a coupling network to further project them into a latent vector. Vector quantization is thus performed on it for discrete representation. Taking advantages of such a compact representation with tractable orders, vanilla transformers are adopted to auto-repressively learn shape distributions. Furthermore, we can freely switch from unconditional generation to conditional generation by concatenating various conditions, such as point clouds, categories and images.

explodes as r grows. *Second, the order of* $\mathbf{z}^v$ *affects the generation quality*. Each grid is highly coupled with neighbors, simply flattening (say, along x-y-z axes) may cause 'ambiguity', leading to sub-optimal generation quality.

3.2. Improved Discrete Representation Learning

One possible solution to solve the first limitation is applying vector quantization in spatial grids instead of volumetric grids inspired by [3]. Specifically, after obtaining point cloud features $f^p \in \mathbb{R}^{n\times c}$, we first project points onto three axis-aligned orthogonal planes. Features of points falling into the same plane grid are aggregated via summation, resulting in three feature maps for the three planes $\{f^{xy}, f^{yz}, f^{xz}\} \in \mathbb{R}^{l\times w\times c}$. Next, the vector quantization is applied to the three planes separately. The primary advantage of tri-planar representation is efficient and compact. It can dramatically reduce the number of grids from $\mathcal{O}(r^3)$ to $\mathcal{O}(r^2)$ while preserving essential 3D information. However, it still suffers from the the ambiguity of order, or worse, since it involves the flattening order of three planes and the order of entries of each plane.

To this end, we further introduce a projection by learning a higher latent space for features of the three planes. This is simply achieved by first concatenating three planes with arbitrary order and then feeding them into a coupling network. Finally, the output is flattened as a projected latent vector, formulated as,

$$f = \tau\left(\mathcal{G}\left(\left[f^{xy};\ f^{yz};\ f^{xz}\right];\ \theta\right)\right) \in \mathbb{R}^{m\times d} \qquad (2)$$

where $[\cdot;\cdot]$ denotes the concatenation operation; $\mathcal{G}(\cdot;\theta)$ is a series of convolution layers with parameters θ; $\tau(\cdot)$ means the operation of flatten with row-major order; m and d indicate the length of latent vector and feature dimension. By applying discrete representation learning in the latent vector, we can describe each 3D shape with $\mathbf{z} = \mathcal{Q}(f)$, where $\mathcal{Q}(\cdot)$ represents vector quantization in Eq. 1.

Remark. Different from existing works [23, 45] that rely on structure design and training strategies in second stage to address the problem of ambiguous order, we tackle it by learning the coupling relationship of spatial grids in the first stage with the help of the second projection. By stacking convolution layers, we increase the receptive field of each element in the latent vector. Additionally, since the features of each spatial grid on the three planes are fused and highly encoded, each element does not have a definite position mapping in 3D space, which results in a tractable order for auto-regression. More in-depth discussions can be found in Sec. 4.6 and the supplementary materials.

Training Objective. We optimize parameters with reconstruction loss. After getting discrete representation $\mathbf{z}$ which represents indices of entries in the codebook, we retrieve the corresponding codes with indices, denoted as $\mathbf{q}_{(\mathbf{z})}$. Subsequently, a decoder with symmetric structures of the encoder are designed to decode $\mathbf{q}_{(\mathbf{z})}$ back to features of the three planes *. Given sampling points $x \in \mathbb{R}^3$, we query their features by projecting them onto each of the three feature planes and performing bilinear interpolation. Features from three planes are accumulated and fed into an implicit function to predict their occupancy values. Finally, we apply binary cross-entropy loss between the predicted values y_o and the ground-truth ones $\tilde{y_o}$,

$$\mathcal{L}_{occ} = -\left(\tilde{y_o}\cdot\log\left(y_o\right) + \left(1-\tilde{y_o}\right)\cdot\log\left(1-y_o\right)\right) \qquad (3)$$

To further train the codebook, we encourage pulling the

*For more details about the first stage architecture, please refer to the Supplementary.

distance between features before and after the vector quantization. Thus, the codebook loss is derived as,

$$\mathcal{L}_{code} = \beta||\text{sg}\left[f\right] - \mathbf{q}_{(\mathbf{z})}||_2^2 + ||f - \text{sg}\left[\mathbf{q}_{(\mathbf{z})}\right]||_2^2 \quad (4)$$

where $\text{sg}\left[\cdot\right]$ denotes the stop-gradient operation [38] and we set $\beta = 0.4$ by default. In sum, the overall loss for the first stage is $\mathcal{L}_{rec} = \mathcal{L}_{occ} + \mathcal{L}_{code}$.

3.3. Learning Priors with Vanilla Transformers

Benefiting from a compact composition and tractable order of discrete representation, models in the second stage can absorbedly learn the correlation between discrete codes, effectively exploring priors of shape composition. We thus adopt a vanilla decoder-only transformer [11] without any specific-designed module.

For unconditional generation, given discretized indices of latent vector $\mathbf{z} = \{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_m\}$, we feed them into a learnable embedding layer to retrieve features with discrete indices [†]. Then, the transformer with multi-head self-attention mechanism predicts the next possible index by learning the distribution of previous indices, $p\left(\mathbf{z}_i \mid \mathbf{z}_{<i}\right)$. This gives the joint distribution of full representation as,

$$p\left(\mathbf{z}\right) = \prod_{i=1}^{m} p\left(\mathbf{z}_i \mid \mathbf{z}_{<i}\right) \quad (5)$$

For conditional generation, users often expect to control the generation process by providing additional conditions. Instead of designing complex modules or training strategies, we simply learn joint distribution given conditions $\mathbf{c}$ by prepending it to $\mathbf{z}$. Equation 5 is thus extended as,

$$p\left(\mathbf{z}\right) = \prod_{i=1}^{m} p\left(\mathbf{z}_i \mid \mathbf{c},\ \mathbf{z}_{<i}\right) \quad (6)$$

where $\mathbf{c}$ denotes a feature vector of given conditions. The simplicity of our model gives the flexibility to learn conditions of any form. Specifically, for 3D conditions such as point clouds, we use our discrete representation learning in Sec. 3.2 to transform them into a vector. As for 2D/1D conditions such as images and classes, we either adopt pre-trained models or embedding layers to extract their features.

Objective. To train second stage, we minimize negative log-likelihood of Eq. 5 or 6 as $\mathcal{L}_{nll} = \mathbb{E}_{x\sim p(x)}\left[-\log p\left(\mathbf{z}\right)\right]$, where $p\left(x\right)$ is the distribution of real data.

Inference. With both models trained on two stages, we use Eq. 5 or 6 to perform shape generation by progressively sampling the next index with top-*k* sampling strategy, until all elements in $\mathbf{z}$ are completed. Then, we feed $\mathbf{q}_{(\mathbf{z})}$ into the decoder of the first stage, and query probabilities of occupancy values for all sampled 3D positions (*e.g.,* 128^3). The output shapes are extracted with Marching Cubes [18].

[†]We reuse the symbol of $\mathbf{z}$ after embedding for simplicity

4. Experiments

This section starts with comparing results on unconditional 3D shape generation, showing more faithful and diverse shapes synthesized by our approach in Sec. 4.1. Next, we show extensive studies on four generation tasks, demonstrating the powerful and flexible ability of ImAM (in Sec. 4.2 $\sim$ 4.5). Lastly, we provide in-depth studies to evaluate the efficacy of our modules and show generalization to real-world data and zero-shot generation (in Sec. 4.6). For all experiments, if necessary, we sample point clouds from output meshes with Poisson Disk Sampling, or reconstruct meshes with our auto-encoder from output points, which is better than Poisson Surface Reconstruction [16]. Please refer to the Supplementary for more details about implementations and qualitative results.

4.1. Unconditional Shape Generation

Data. We consider ShapeNet [4] as our main dataset for generation, following previous literature [14, 8, 36]. We use the same training split and evaluation setup from [14] for fair comparability. Five categories of car, chair, plane, rifle and table are used for testing. As ground-truth, we extract mesh from voxelized models with 256^3 resolution in [13].

Baselines. We compare ImAM with five state-of-the-art models, including GAN-based IM-GAN[8] and GBIF [14], flow-based PointFlow [46], score-based ShapeGF [2] and diffusion-based PVD [51]. We train these methods on the same data split with the official implementation.

Metrics and Settings. The size of the generated set is 5 times the size of the test set, the same as [8, 14]. As suggested by [8], we use the Light Field Descriptor (LFD) [6] as our primary similarity distance metric between two shapes. Coverage (COV) [1], Minimum Matching Distance (MMD) [1] and Edge Count Difference (ECD) [14] are adopted to evaluate the diversity, fidelity and overall quality of synthesized shapes. We also use 1-Nearest Neighbor Accuracy (1-NNA) [46] (with Chamfer Distance) to measure the distribution similarity. The number of sampled points is 2048. Besides, it is well known that COV does not penalize outliers. To rule out the false positive coverage, we introduce CovT, counting as match between a generation and ground truth shape only if LFD between them is smaller than a threshold *t*. In practice, *t* could vary across different categories based on the scale and complexity of the shape, and we empirically use mean MMD as the threshold and found it effective in identifying correct matches.

Results Analysis. Results are reported in Table 1. First, ImAM achieves state-of-the-art performance with regard to ECD and 1-NNA. It significantly demonstrates the superiority of our model over synthesizing high-quality shapes. We notice that the result of Car is not good on the metric of 1-NNA. One possible reason is that ImAM tries to generate meshes of tires and seats inside cars, which may not

Metrics	Methods	Categories					Avg
		Plane	Car	Chair	Rifle	Table	
ECD ↓	IM-GAN [8]	923	3172	658	371	418	1108
	GBIF [14]	945	2388	354	195	411	858
	PointFlow [46]	2395	5318	426	2708	3559	2881
	ShapeGF [2]	1200	2547	443	672	114	915
	PVD [51]	6661	7404	1265	3443	745	3904
	Ours	**236**	**842**	**27**	**65**	**31**	**240**
1-NNA ↓	IM-GAN [8]	78.18	89.39	65.83	69.38	65.31	73.62
	GBIF [14]	80.22	87.19	63.95	66.98	60.96	71.86
	PointFlow [46]	73.61	74.75	70.18	64.77	74.81	71.62
	ShapeGF [2]	74.72	62.81	59.15	60.65	55.58	62.58
	PVD [51]	81.09	**57.37**	62.36	77.32	74.31	70.49
	Ours	**59.95**	76.58	**57.31**	**57.28**	**54.76**	**61.17**
COV ↑	IM-GAN [8]	77.01	65.37	76.38	73.21	85.71	75.53
	GBIF [14]	**80.96**	**78.85**	**80.95**	**77.00**	85.13	**80.57**
	PointFlow [46]	65.64	64.97	57.49	48.52	71.95	61.71
	ShapeGF [2]	76.64	71.85	79.41	70.67	**87.54**	77.22
	PVD [51]	58.09	58.64	68.93	56.12	76.84	63.72
	Ours	79.11	73.25	80.81	74.26	84.01	78.29
CovT ↑	IM-GAN [8]	41.03	50.63	45.68	51.68	46.50	47.10
	GBIF [14]	32.38	52.76	39.77	50.00	43.68	43.72
	PointFlow [46]	35.85	47.76	28.48	34.81	30.98	35.57
	ShapeGF [2]	40.17	53.63	43.69	51.05	**48.50**	47.41
	PVD [51]	12.11	43.36	38.82	33.33	43.68	34.26
	Ours	**45.12**	**56.64**	**49.82**	**55.27**	48.03	**50.98**
MMD ↓	IM-GAN [8]	3418	1290	2881	3691	2505	2757
	GBIF [14]	3754	1333	3015	3865	2584	2910
	PointFlow [46]	3675	1393	3322	4038	2936	3072
	ShapeGF [2]	3530	1307	2880	3762	2420	2780
	PVD [51]	4376	1432	3064	4274	2623	3154
	Ours	**3124**	**1213**	**2703**	**3628**	**2374**	**2608**

Table 1: Results of unconditional generation. Models are trained for each category. The best and second results are highlighted in **bold** and underlined.

be very friendly to CD. Second, our model has a clear advantage on both MMD and CovT metrics compared with all competitors, which separately indicates the outstanding fidelity and diversity of our generated shapes. Third, though GBIF achieves relatively good results on COV, it gets worse results on CovT, suggesting that most of the matched samples come from false positive pairs. ImAM, on the contrary, gets second best performance on COV, but higher than GBIF on CovT by about 6 points, showing that our generated shapes enjoy the advantage of high-quality and fewer outliers. Lastly, we visualize shapes with multiple categories in Fig. 4, further supporting the quantitative results and conclusions described above.

4.2. Class-guide Generation

We first evaluate the versatility of ImAM on class-guide generation, which requires generating shapes given a category label. It is a basic conditional generation task. We use the same dataset and evaluation metrics as in Sec. 4.1.

Baselines. We choose two recently-published works as competitors due to the similar motivation. One is two-stage generative model GBIF [14], the other is AR method AutoSDF [23]. We simply modify mask-condition of [14] to class-condition, and additionally add class token to transformers for [23] which is the same as ImAM.

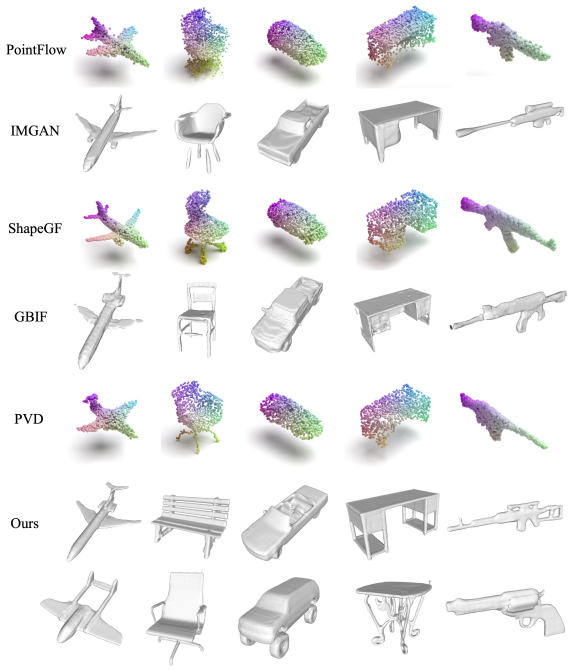

Figure 4: Qualitative results of unconditional generation.

Results Analysis. As shown in Tab. 2, ImAM outperforms both competitors across 5 categories by a significant margin, achieving state-of-the-art results on all metrics. Particularly, it gets advantages on the metric of 1-NNA, strongly demonstrating the versatility of ImAM on class-guide generation. Qualitative results of 5 different categories are further illustrated in Fig. 5. As observed, the generated quality of our method is clearly better than GBIF and AutoSDF, while preserving more diversity in types and shapes.

4.3. Multi-modal Partial Point Completion

We further verify the ability of our model in conditional generation by giving partial point clouds. Here, we advocate the multi-modal completion, since there are many possibilities of the completed shape given the partial shape. It is the essence of generative model, where being faithful to the given partial conditions but using your imagination.

Data. We use ShapeNet dataset for testing. Two settings are considered here, (1) perspective completion [49]: randomly sampling a viewpoint and then removing the $25\% \sim 75\%$ furthest points from the viewpoint; (2) bottom-half completion [23]: removing all points from the top half of shapes.

Baselines. Four multi-modal completion models are chosen as baselines, including one generative adversarial model cGAN [42], one diffusion model PVD [51], two AR models ShapeFormer [45] and AutoSDF [23].

Metrics	Methods	Categories					AVG
		Plane	Car	Chair	Rifle	Table	
COV ↑	GBIF [14]	68.72	69.64	75.94	68.98	81.72	73.00
	AutoSDF [23]	70.46	52.77	63.25	48.10	72.19	61.35
	Ours	**81.58**	**71.58**	**83.98**	**75.74**	**85.48**	**79.67**
CovT ↑	GBIF [14]	24.10	38.63	32.69	35.44	37.80	33.73
	AutoSDF [23]	30.66	40.49	31.00	34.60	36.10	34.57
	Ours	**56.49**	**52.70**	**45.09**	**52.74**	**49.32**	**51.27**
MMD ↓	GBIF [14]	4736	1479	3220	4246	2763	3289
	AutoSDF [23]	3706	1456	3249	4115	2744	3054
	Ours	**3195**	**1285**	**2871**	**3729**	**2430**	**2702**
ECD ↓	GBIF [14]	1327	2752	1589	434	869	1394
	AutoSDF [23]	1619	4256	1038	1443	462	1764
	Ours	**571**	**1889**	**419**	**196**	**285**	**672**
1-NNA ↓	GBIF [14]	91.47	92.43	75.61	83.12	70.19	82.56
	AutoSDF [23]	83.31	87.76	69.34	77.43	67.20	77.01
	Ours	**66.81**	**83.39**	**64.83**	**57.28**	**59.55**	**66.37**

Table 2: Results of class-guide generation. Models are trained on 13 categories of ShapeNet.

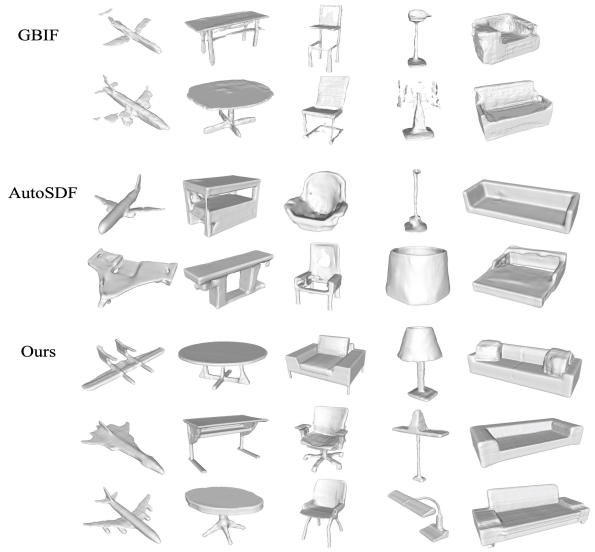

Figure 5: Qualitative results of class-guide generation.

Metrics and Settings. We complete 10 samples for 100 randomly selected shapes of three categories, *i.e.*, chair, sofa and table. Following [42], we use Total Mutual Difference (TMD) to measure the diversity. Minimum Matching Distance [1] (MMD) with Chamfer Distance and Unidirectional Hausdorff Distance (UHD) [32] are adopted to measure the faithfulness of completed shapes.

Results Analysis. We first report perspective completion results in Tab. 3. ImAM beats all baselines and achieves state-of-the-art performance. Importantly, we outperform Shapeformer on all classes and metrics, which also utilizes an AR model with transformers to learn shape distribution. On the other hand, we compare with AutoSDF in its bottom-half completion setting. Results from Tab. 4 illustrates that ImAM outperforms AutoSDF in terms of TMD and MMD. It strongly suggests the flexibility and versatility of our proposed method. Qualitative results in Fig. 6 show the diversity and fidelity of our completed shapes.

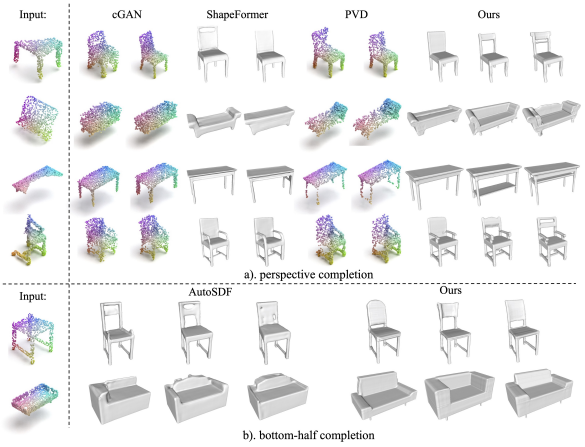

Figure 6: Qualitative results of partial point completion.

Metrics	Methods	Categories			AVG
		Chair	Sofa	Table	
TMD ↑ ($\times 10^2$)	cGAN [42]	1.708	0.687	**1.707**	1.367
	PVD [51]	1.098	0.811	0.839	0.916
	ShapeFormer [45]	1.159	0.698	0.677	0.845
	Ours	**2.042**	**1.221**	1.538	**1.600**
UHD ↓ ($\times 10^2$)	cGAN [42]	7.836	7.047	9.406	8.096
	PVD [51]	10.79	13.88	11.38	12.02
	ShapeFormer [45]	6.884	8.658	6.688	7.410
	Ours	**6.439**	**6.447**	**5.948**	**6.278**
MMD ↓ ($\times 10^3$)	cGAN [42]	1.665	1.813	1.596	1.691
	PVD [51]	2.352	2.041	2.174	2.189
	ShapeFormer [45]	1.055	1.100	1.066	1.074
	Ours	**0.961**	**0.819**	**0.828**	**0.869**

Table 3: Results of multi-modal partial point completion. The missing parts vary according to random viewpoints.

Metrics	Methods	Categories			AVG
		Chair	Sofa	Table	
TMD ↑ ($\times 10^2$)	AutoSDF [23]	2.046	1.609	3.116	2.257
	Ours	**3.682**	**2.673**	**10.30**	**5.552**
UHD ↓ ($\times 10^2$)	AutoSDF [23]	**6.793**	9.950	**8.122**	8.289
	Ours	6.996	**6.599**	10.87	**8.155**
MMD ↓ ($\times 10^3$)	AutoSDF [23]	1.501	1.154	**2.600**	**1.751**
	Ours	**1.477**	**1.127**	2.906	1.837

Table 4: Quantitative results of multi-modal partial point completion. The missing parts are always the top half.

4.4. Image-guide Generation

Next, we show the flexibility that ImAM can easily extend to image-guide generation, which is a more challenging task. The flexibility lies in that (1) it is implemented by the easiest way of feature concatenation; (2) the condition form of images is various, which could be 1-D feature

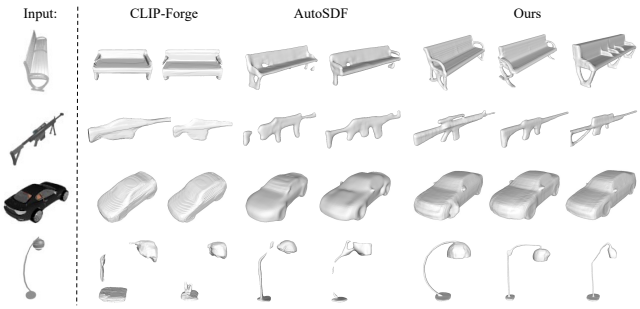

Figure 7: Visualizations of image-guide shape generation.

vector or patch tokens, or 2-D feature maps. For simplicity, we use a pretrained CLIP model (*i.e.*, ViT-B/32) to extract feature vectors of images as conditions. All experiments are conducted on ShapeNet dataset with rendered images.

Baselines. CLIP-Forge [31] is a flow-based model, which is trained with pairs of images and shapes. We take it as our primary baseline for two reasons: (1) it is a generation model instead of reconstruction model, and (2) it originally uses CLIP models to extract image features.

Metrics and Settings. We evaluate models with the test split of 13 categories. For each category, we randomly sample 50 singe-view images and then generate 5 shapes for evaluation. As a generation task, TMD is adopted to measure the diversity. We further use MMD with Chamfer Distance and Fréchet Point Cloud distance (FPD) [32] to measure the fidelity compared with the ground truth.

Results Analysis. Results are reported in Tab. 5. ImAM wins out in terms of both fidelity and diversity. In particular, we achieve a great advantage on both TMD and FPD, demonstrating the effectiveness of ImAM applied to image-guide generation. It is also successfully echoed by qualitative visualizations in Fig. 7. Our samples are diverse and appear visually faithful to attributes of the object in images.

4.5. Text-guide Generation

Encouraged by promising results on image-guide generation, we also turn ImAM to text-to-shape generation. The same pretrained CLIP model is used to extract single embedding for text conditions. Note that we did it on purpose, not using word sequence embedding, but only using a single features vectors in our model to show its efficacy and scalability to the simplest forms of conditions.

Data. To our knowledge, the only existing largest paired text-shape dataset is Text2Shape [7], which provides language descriptions for two objects from ShapeNet, *i.e.*, chair and table. Thereby, we consider it as our main dataset to perform text-guide shape generation.

Baselines. We compare our model with two state-of-the-arts text-to-shape generation model. One is CLIP-Forge [31] under the supervised learning setting, using the same condition extractor as ours. The other is ITG [17], which adopts BERT to encode texts into sequence embeddings.

METHOD	TMD ($\times 10^2$) ↑	MMD ($\times 10^3$) ↓	FPD ↓
AutoSDF [23]	2.523	**1.383**	2.092
Clip-Forge [31]	2.858	1.926	8.094
Ours	**4.274**	1.590	**1.680**

Table 5: Quantitative results of image-guide generation.

METHOD	TMD ($\times 10^1$) ↑	MMD ($\times 10^3$) ↓	Acc ↑
ITG [17]	N/A	2.187	29.13
AutoSDF [23]	0.342	2.165	36.95
CLIP-Forge [31]	0.400	2.136	53.68
Ours	**0.565**	**1.846**	**59.93**

(a) Descriptions as text queries

METHOD	TMD ($\times 10^1$) ↑	FPD ↓	Acc. ↑
AutoSDF [23]	0.752	5.84	41.09
CLIP-Forge [31]	0.961	**4.14**	55.00
Ours	**1.246**	5.39	**60.87**

(b) Prompts as text queries

Table 6: Quantitative results of text-guide generation.

Metrics and Settings. All models are trained on the train split of Text2Shape dataset with their official codes. We evaluate our model with two types of text queries: (1) description: the test split of Text2Shape; (2) prompts: customized short phrases provided by [31] containing attributes for chair and table. We additionally use Accuracy (Acc.) [31] to measure the fidelity. The Accuracy is calculated by a pretrained PointNet [27] classifier on ShapeNet.

Results Analysis. We report quantitative results of text-guide generation in Tab. 6. ImAM achieves promising results on TMD, MMD and Acc, showing good generalization performance across different conditional generation tasks.

4.6. Ablation Study

Lastly, we provide in-depth studies to dissect the efficacy of our ImAM framework, and several proposed designs. More discussions are provided in the Supplementary.

Design Choices in Discrete Representation Learning. As a key contribution of this paper, we first discuss the efficacy of our improved discrete representation learning. We use 'Vector' to denote our design since we apply vector quantization to latent vector. Similarly, 'Grid' and 'Tri-Plane' refer to baselines applying vector quantization to volumetric grids and the three planes, respectively. Results in Tab. 7 show that 'Grid' gets better IoU performance for shape reconstruction in the first stage, but fails to train transformers in the second stage due to extreme long length of sequence (*e.g.*, 32^3). In contrast, our model not only achieves comparable reconstruction results (#0 *vs.* #2), but also outperforms 'Tri-Plane' by a large margin on generation quality (#1 *vs.* #2). The latter shows inferior results due to 'ambiguity of order' (see Suppl.) It significantly proves the efficacy

Figure 8: Results of real-world image-guide generation. Images are randomly selected from internet.

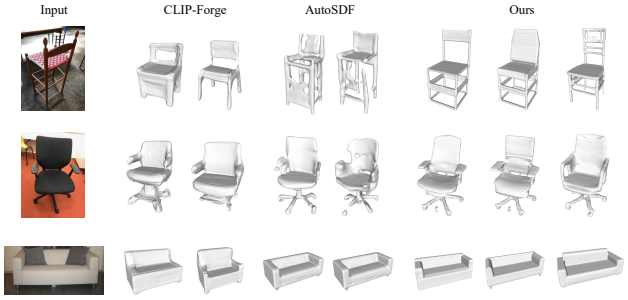

Figure 9: Results of real-world image-guide generation. Samples are from Pix3D dataset.

of our proposed coupling network, improving the plasticity and flexibility of the model. We also explore different design choices in Tab. 7. By gradually increasing feature resolutions or the number of entries in the codebook, we achieve better performance on IoU. This observation is consistent with [48], as the capacity of the discrete representation is affected by these two factors. We do not try larger parameters, as it would significantly increase the computation cost.

Image-guide Generation in Real-world. We further investigate the generalizability of our model on real-world images. We use the model trained on ShapeNet as described in Sec. 4.4, and download images from internet as conditions. Figure 8 shows the qualitative results for three categories, *i.e.*, plane, chair and table. Our model sensitively capture major attributes of objects in the image and produce shapes faithful to them (see the first column to the left). Meanwhile, our synthesized samples enjoy the advantage of diversity by partially sticking to the images.

Figure 9 also show results of our model on Pix3D dataset (trained on ShapeNet, without any finetuning). Compared with other competitors, ImAM is capable of generating high-quality and realistic shapes that highly match the shape of objects in real-world images. It significantly highlights the strong generalization ability of our method.

Figure 10: Results of zero-shot text-to-shape generation.

NUM.	TYPE	#ENTRY	RESO.	STAGE 1 IoU ↑	STAGE 2 1-NNA / ECD ↓
0	Grid	4096	32	88.87	×
1	Tri-Plane			87.81	73.67 / 743
2	Vector			88.01	**59.95 / 236**
3		4096	16	79.17	-
4		2048	32	86.99	
5		1024		86.57	

Table 7: Ablation study of auto-encoder design choices. We report 1-NNA/ECD for plane category. 'RESO.' means the resolution of feature map for vector quantization. '×': cannot report due to extreme memory cost. '-': not report. ***Notably, without the coupling network, our method naturally degenerates into 'Tri-Plane' representation.***

Zero-shot Text-to-shape Generation. Inspired by CLIP-Forge [31], we utilize the CLIP model to achieve zero-shot text-to-shape generation. At training, we only use the rendered images of 3D shapes. At inference, we substitute image features with text features encoded by the CLIP model. Figure 10 shows our ability of zero-shot generation, where shape attributes are controlled with different prompts.

5. Conclusion

We introduce an improved AR model for 3D shape generation. By projecting volumetric grids of encoded input shapes onto three axis-aligned orthogonal feature planes, which are then coupled into a latent vector, we reduce computational costs and create a more tractable order for AR learning. Our compact and tractable representations enable easy switching between unconditional and conditional generation with multi-modal conditioning inputs. Extensive experiments show that our model outperforms previous methods on multiple generation tasks.

Acknowledgements. This work is supported by China Postdoctoral Science Foundation (2022M710746). Yanwei Fu is with the School of Data Science, Fudan University, and Fudan ISTBI—ZJNU Algorithm Centre for Brain-inspired Intelligence, Zhejiang Normal University, Jinhua, China.

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. 2, 5, 7

[2] Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. Learning gradient fields for shape generation. In *European Conference on Computer Vision*, pages 364–381. Springer, 2020. 5, 6

[3] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. *arXiv preprint arXiv:2112.07945*, 2021. 4

[4] 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. 5

[5] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. *arXiv preprint arXiv:2202.04200*, 2022. 2

[6] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, volume 22, pages 223–232. Wiley Online Library, 2003. 5

[7] Kevin Chen, Christopher B Choy, Manolis Savva, Angel X Chang, Thomas Funkhouser, and Silvio Savarese. Text2shape: Generating shapes from natural language by learning joint embeddings. In *Asian conference on computer vision*, pages 100–116. Springer, 2018. 8

[8] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019. 2, 5, 6

[9] An-Chieh Cheng, Xueting Li, Sifei Liu, Min Sun, and Ming-Hsuan Yang. Autoregressive 3d shape generation via canonical mapping. *arXiv preprint arXiv:2204.01955*, 2022. 3

[10] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016. 2

[11] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883, 2021. 2, 3, 5

[12] Haoqiang Fan, Hao Su, and Leonidas 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*, pages 605–613, 2017. 2

[13] Christian Häne, Shubham Tulsiani, and Jitendra Malik. Hierarchical surface prediction for 3d object reconstruction. In *2017 International Conference on 3D Vision (3DV)*, pages 412–420. IEEE, 2017. 2, 5

[14] Moritz Ibing, Isaak Lim, and Leif Kobbelt. 3d shape generation with grid-based implicit functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13559–13568, 2021. 2, 5, 6, 7

[15] Chiyu Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, Thomas Funkhouser, et al. Local implicit grid representations for 3d scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6001–6010, 2020. 2

[16] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006. 5

[17] Zhengzhe Liu, Yi Wang, Xiaojuan Qi, and Chi-Wing Fu. Towards implicit text-guided 3d shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17896–17906, 2022. 8

[18] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987. 2, 5

[19] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015. 2

[20] Oier Mees, Maxim Tatarchenko, Thomas Brox, and Wolfram Burgard. Self-supervised 3d shape and viewpoint estimation from single images for robotics. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6083–6089. IEEE, 2019. 2

[21] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019. 2

[22] Mateusz Michalkiewicz, Jhony K Pontes, Dominic Jack, Mahsa Baktashmotlagh, and Anders Eriksson. Implicit surface representations as layers in neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4743–4752, 2019. 2

[23] 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. 2, 3, 4, 6, 7, 8

[24] Charlie Nash, Yaroslav Ganin, SM Ali Eslami, and Peter Battaglia. Polygen: An autoregressive generative model of 3d meshes. In *International Conference on Machine Learning*, pages 7220–7229. PMLR, 2020. 3

[25] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 2

[26] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *European Conference on Computer Vision*, pages 523–540. Springer, 2020. 2

[27] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2, 8

[28] Xuelin Qian, Li Wang, Yi Zhu, Li Zhang, Yanwei Fu, and Xiangyang Xue. Impdet: Exploring implicit fields for 3d object detection. *arXiv preprint arXiv:2203.17240*, 2022. 2

[29] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019. 3

[30] Gernot Riegler, Ali Osman Ulusoy, Horst Bischof, and Andreas Geiger. Octnetfusion: Learning depth fusion from data. In *2017 International Conference on 3D Vision (3DV)*, pages 57–66. IEEE, 2017. 2

[31] Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshan. Clip-forge: Towards zero-shot text-to-shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18603–18613, 2022. 8, 9

[32] Dong Wook Shu, Sung Woo Park, and Junseok Kwon. 3d point cloud generative adversarial network based on tree structured graph convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3859–3868, 2019. 7, 8

[33] Vincent Sitzmann, Eric Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. *Advances in Neural Information Processing Systems*, 33:10136–10147, 2020. 2

[34] Jonathan Dyssel Stets, Yongbin Sun, Wiley Corning, and Scott W Greenwald. Visualization and labeling of point clouds in virtual reality. In *SIGGRAPH Asia 2017 Posters*, pages 1–2. 2017. 2

[35] Yongbin Sun, Sai Nithin R Kantareddy, Rahul Bhattacharyya, and Sanjay E Sarma. X-vision: An augmented vision tool with real-time sensing ability in tagged environments. In *2018 ieee international conference on rfid technology & application (rfid-ta)*, pages 1–6. IEEE, 2018. 2

[36] Yongbin Sun, Yue Wang, Ziwei Liu, Joshua Siegel, and Sanjay Sarma. Pointgrow: Autoregressively learned point cloud generation with self-attention. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 61–70, 2020. 3, 5

[37] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29, 2016. 3

[38] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 5

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017. 3

[40] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions On Graphics (TOG)*, 36(4):1–11, 2017. 2

[41] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *Advances in neural information processing systems*, 29, 2016. 2

[42] 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, 7

[43] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9621–9630, 2019. 2

[44] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 2

[45] Xingguang Yan, Liqiang Lin, Niloy J Mitra, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Shapeformer: Transformer-based shape completion via sparse representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6239–6249, 2022. 2, 3, 4, 6, 7

[46] 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, 6

[47] Jianglong Ye, Yuntao Chen, Naiyan Wang, and Xiaolong Wang. Online adaptation for implicit object tracking and shape reconstruction in the wild. *arXiv preprint arXiv:2111.12728*, 2021. 2

[48] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*, 2021. 9

[49] 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

[50] Long Zhao, Zizhao Zhang, Ting Chen, Dimitris Metaxas, and Han Zhang. Improved transformer for high-resolution gans. *Advances in Neural Information Processing Systems*, 34, 2021. 2

[51] 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, 6, 7