

Make-A-Shape: a Ten-Million-scale 3D Shape Model

Ka-Hei Hui^{*1}Aditya Sanghi^{*2}Arianna Rampini²Kamal Rahimi Malekshan²Zhengzhe Liu¹Hooman Shayani²Chi-Wing Fu¹^{*}These authors contributed equally.¹The Chinese University of Hong Kong²Autodesk AI Lab<https://edward1997104.github.io/make-a-shape/>

Figure 1. *Make-A-Shape* is a large 3D generative model trained on over 10 millions diverse 3D shapes. As demonstrated above, it exhibits the capability of unconditionally generating a large variety of 3D shapes over a wide range of object categories, featuring intricate geometric details, plausible structures, nontrivial topologies, and clean surfaces.

Abstract

Significant progress has been made in training large generative models for natural language and images. Yet, the advancement of 3D generative models is hindered by their substantial resource demands for training, along with inefficient, non-compact, and less expressive representations. This paper introduces *Make-A-Shape*, a new 3D generative model designed for efficient training on a vast scale, capable of utilizing 10 millions publicly-available shapes. Technical-wise, we first innovate a wavelet-tree representation to compactly encode shapes by formulating the sub-band coefficient filtering scheme to efficiently exploit coefficient relations. We then make the representation generatable by a diffusion model by devising the subband coefficients packing scheme to layout the representation in a low-resolution grid. Further, we derive the subband adaptive training strategy to train our model to effectively learn to generate coarse and detail wavelet coefficients. Last, we extend our framework to be controlled by additional input conditions to enable it to generate shapes from assorted modalities, e.g., single/multi-view images, point clouds, and

low-resolution voxels. In our extensive set of experiments, we demonstrate various applications, such as unconditional generation, shape completion, and conditional generation on a wide range of modalities. Our approach not only surpasses the state of the art in delivering high-quality results but also efficiently generates shapes within a few seconds, often achieving this in just 2 seconds for most conditions. Our source code is available at <https://github.com/AutodeskAILab/Make-a-Shape>.

1. Introduction

Large-scale generative models have increasingly become highly capable of generating realistic outputs [72, 74, 75, 101], leading to numerous commercial applications in design, marketing, e-commerce, and more. This success has led to the question of whether it is possible to develop a large-scale 3D generative model, which could potentially exhibit intriguing emergent properties and facilitate a variety of applications. However, most current 3D generative models have lagged behind, being either limited in quality, focused on small 3D datasets [6, 18, 28, 32, 56, 82, 100,

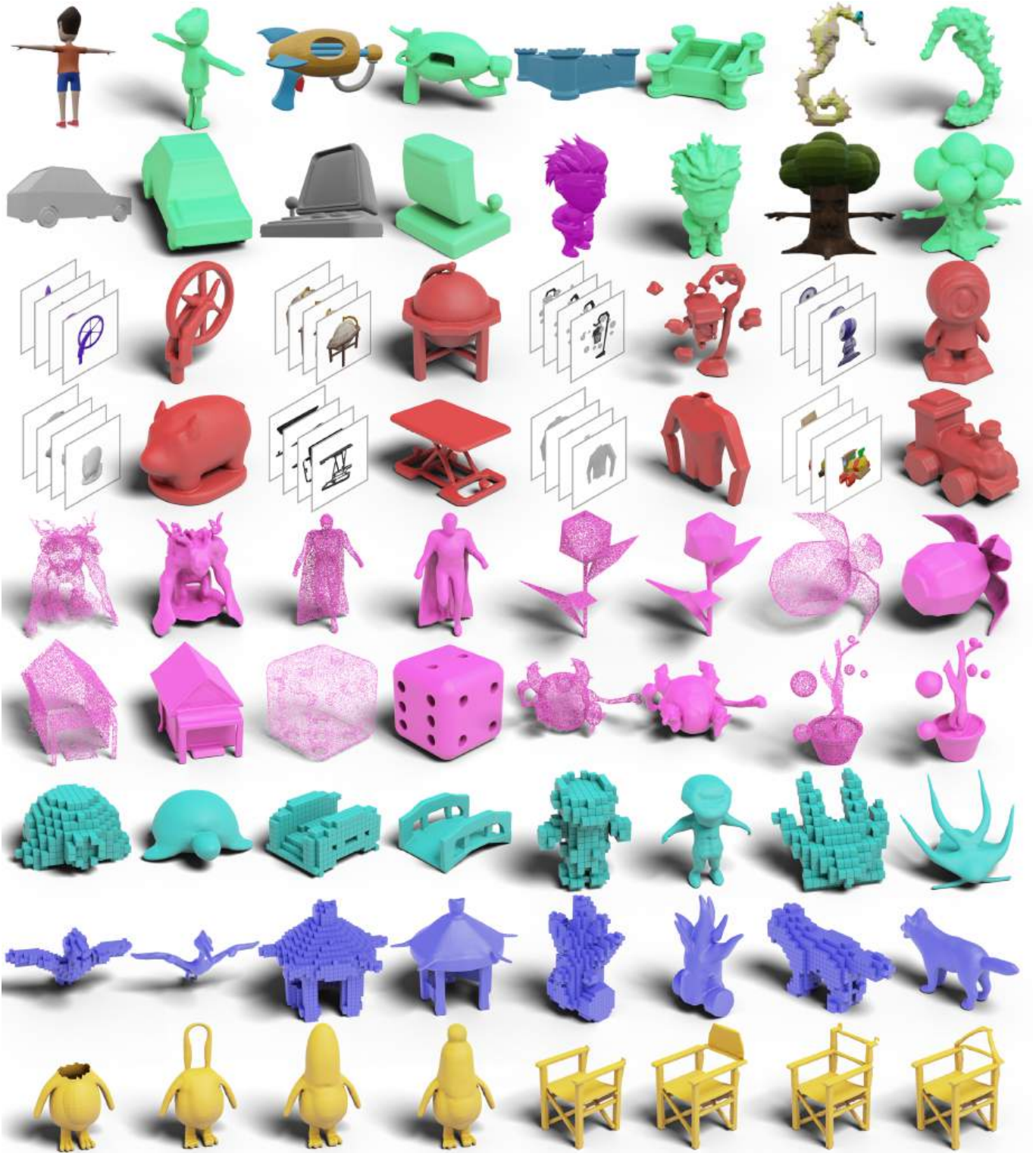


Figure 2. Make-A-Shape is able to generate a large variety of shapes for diverse input modalities: single-view images (rows 1 & 2), multi-view images (rows 3 & 4), point clouds (rows 5 & 6), voxels (rows 7 & 8), and incomplete inputs (last row). The resolution of the voxels in rows 7 & 8 are 16^3 and 32^3 , respectively. In the top eight rows, odd columns show the inputs whereas even columns show the generated shapes. In the last row, columns 1 & 4 show the partial input whereas the remaining columns show the diverse completed shapes.

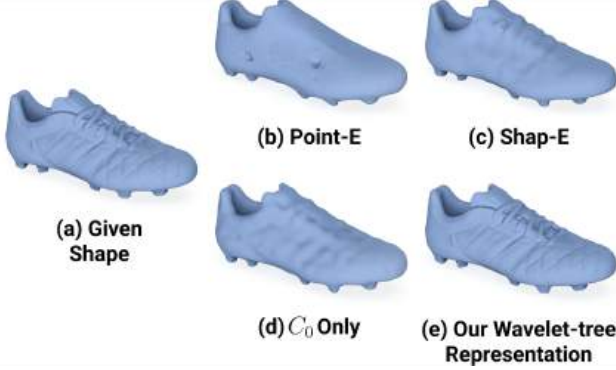


Figure 3. Reconstructing the SDF of a shape (a) using different methods: (b) Point-E [62], (c) Shap-E [33], (d) coarse coefficients C_0 [28], and (e) our wavelet-tree representation. Our approach (e) can more faithfully reconstruct the shape’s structure and details.

105] or allowing a single condition [25, 33, 40, 43, 62, 98].

Training large generative models in 3D, compared to 2D images, faces several significant challenges. First, having an extra spatial dimension in 3D substantially increases the number of input variables that require a neural network to model, resulting far more network parameters. This is particularly evident in U-Net-based diffusion models [24, 83, 84], which generate memory-intensive feature maps that are often too large for GPUs to process, thus prolonging the training time [26]. Second, scaling a generative model to 3D introduces data handling complexities not present with 2D images. Most storage and data handling for training large models takes place on cloud services such as AWS or Azure. 3D data escalates the cost of storage and time to download the data in each training iteration. Third, there are many ways to represent 3D shapes. It remains unclear which one best achieves high representation quality while maintaining a good representation compactness for efficient training.

Recent large-scale generative models for 3D shapes tackle these issues through two main strategies. The first employs lossy input representations to reduce the number of input variables that the model must process. However, it comes at the expense of omitting important details, thus failing to faithfully capture the shape. Key examples of this strategy are Point-E [62], which utilizes point clouds as input, and Shap-E [33], which uses latent vectors. Their trade-off is evident in Figure 3 and Table 1, where a significant loss of detail is often observed when reconstructing the ground-truth signed distance function (SDF). The second strategy employs multi-view images to represent the geometry [25, 40, 43, 98]. In this approach, a generative network utilizes differentiable rendering to produce images of the generated shape for comparing with ground-truth multi-view images to facilitate learning. These methods gener-

Table 1. Comparing different 3D representations on the GSO dataset [15] in terms of Intersection Over Union (IOU) and processing time. “Extra Network” denotes the need for training multiple networks to obtain SDF; “Process Time” refers to the time required to convert from one representation to SDF; and “Input Variable” reports the number of floating-point numbers adopted in each representation. It is noteworthy that our representation has a similar parameter count as Shap-E [33], yet it does not need an extra network and achieves faster conversion.

Representation	IOU	Input Variables	Extra Network	Process Time
Ground-truth SDF (256^3)	1.0	16777216	No	—
Point-E [62]	0.8642	12288	Yes	~1 second
Shap-E [33]	0.8576	1048576	Yes	~5 minutes
Coarse Component [28]	0.9531	97336	No	~1 second
Wavelet tree (ours)	0.9956	1129528	No	~1 second

ally require extensive training time, as they use differentiable rendering for loss calculation, which can be slow and may not capture full geometry in one training example. Our framework, on average, processes 2x to 6x more training shapes in one day than these methods, despite utilizing a less powerful GPU (A10G vs. A100), as detailed in Table 2.

These issues arise due to the lack of a suitable 3D representation that is expressive, compact, and efficient to learn. In this work, we introduce a new 3D representation, the *wavelet-tree representation*, designed for encoding 3D shapes for large-scale model training. This representation employs a wavelet decomposition on a high-resolution SDF grid to yield a coarse coefficient subband and multiple multiscale detail coefficient subbands. Beyond [28], which discards all detail subbands for efficient generation, we design a family of techniques to enable large model training, considering both coarse and detail coefficients: (i) *subband coefficients filtering* to identify and retain information-rich detail coefficients in the detail subbands, such that our representation can compactly include more shape details; (ii) *subband coefficients packing* to rearrange the wavelet-tree representation in a low-resolution spatial grid, such that the re-arranged representation can become diffusible, *i.e.*, generatable by a diffusion model; and (iii) *subband adaptive training strategy* to enable efficient model training on both coarse and detail coefficients, such that the training can attend to the overall shape and also the important but sparse shape details. Besides, we formulate various conditioning mechanisms to accommodate flexibly input conditions, such as point clouds, voxels, and images. Hence, our new representation, while being compact, can faithfully retain most shape information and facilitate effective training of a large generative model on over millions of 3D shapes.

With the above technical contributions, we can generate a representation that is notably *expressive*, capable of encoding a shape with minimal loss; for example, a 256^3 grid can be bijectively encoded into the wavelet-tree representation in around one second, yet with an IoU of 99.56%.

Table 2. Efficiency comparison with state-of-the-art methods. The results for rows 4-6 are sourced from the concurrent works, DMV3D [98], Instant3D [40], and LRM [25], respectively. For single-view, we present inference time for both 10 and 100 iterations (iter.), with the latter being the optimal hyperparameter for quality, as per our ablation study. For multi-view, 10 iterations is identified as the optimal. For the training time, since different methods use different number of GPUs, we compare their training speed by the number of training shapes that it can process in one day divided by the number of GPUs used. Note that training time is not available for Point-E [62] and Shape-E [33].

Method	Inference time	# Training shapes in 1 day / GPU
Point-E [62]	~ 31 sec	—
Shape-E [33]	~ 6 sec	—
One-2-3-45 [43]	~ 45 sec	~ 50k (A10G)
DMV3D [98]	~ 30 sec	~ 110k (A100)
Instant3D [40]	~ 20 sec	~ 98k (A100)
LRM [25]	~ 5 sec	~ 74k (A100)
Ours (single-view 10 iter.)	~ 2 sec	~ 290k (A10G)
Ours (single-view 100 iter.)	~ 8 sec	
Ours (multi-view 10 iter.)	~ 2 sec	~ 250k (A10G)

Simultaneously, our representation is *compact*, characterized by a low number of input variables. This is almost akin to lossy representations like latent vectors [33], yet it achieves this without necessitating additional training of an autoencoder, while having higher quality, as shown in Table 1. Last, our representation is *efficient*, enabling efficient streaming and training. For instance, streaming and loading a sophisticatedly compressed 256^3 SDF grid takes 266 milliseconds, while our representation requires only 184 milliseconds for the same process. The 44.5% reduction in data loading time is crucial for large-scale model training.

Overall, our generative model can be trained effectively, enabling also fast inference and taking just *few seconds* to generate high-quality shapes, as compared to existing methods reported in Table 2. We name our proposed generation framework *Make-A-Shape*. This framework facilitates the training of an unconditional generative model and extended models under different input conditions on an extensive dataset comprising 10 million 3D shapes over a wide range of object categories. It successfully produces a range of plausible shapes, as illustrated in Figures 1 and 2.

2. Related Work

Neural Shape Representations. In recent years, there has been significant research in integrating deep learning with various 3D representations. Initial works such as [54, 95] focused on volumetric representations for different tasks through the use of 3D convolutional networks. Multiview CNNs were also explored [66, 86], by means of first rendering 3D shapes into multiple images, upon which 2D CNNs are applied for use in downstream applications.

Subsequently, deep learning methods for point clouds were introduced in PointNet [67], and later, additional inductive biases such as convolution were adopted, as seen in [68, 90]. Neural networks can also be employed to represent 3D shapes by predicting either signed distance functions (SDFs) or occupancy fields. These representations, typically known as neural implicit representations, have been a popular subject of exploration [6, 56, 63]. Other explicit representations such as meshes have been explored in works such as [21, 53, 61, 88]. Another prevalent 3D representation, boundary representation (BREP), has only been examined recently in studies such as [31, 32, 38, 93] for discriminative and generative applications.

Recently, some works [28, 47] have investigated the use of wavelets to decompose an SDF signal into multi-scale wavelet coefficients. These methods, however, filter out high-frequency details to enhance learning efficiency, albeit at the expense of shape fidelity. In this work, we introduce a novel representation known as the wavelet-tree representation. We consider both coarse and information-rich detail coefficients to compactly yet nearly losslessly encode 3D shapes. Enabled by various techniques that we shall introduce, our representation enables high-quality shape generation, while remaining compact for scalability across a large 3D data comprising over millions of shapes.

3D Generative Models. The initial efforts in the field of 3D generative models primarily concentrated on Generative Adversarial Networks (GANs) [19, 92]. Subsequently, autoencoders are trained and GANs are then utilized to process the latent spaces of these autoencoders, enabling generative models on representations such as point clouds [1] and implicit representations [6, 29, 106]. More recent studies [2, 18, 79] incorporated GANs with differentiable rendering, where multiple rendered views are employed as the loss signal. There has also been a focus on normalizing flows [35, 76, 100] and Variational Autoencoder (VAE)-based generative models [60]. Autoregressive models have gained popularity in 3D generative modeling and have been extensively explored [8, 59, 61, 77, 87, 99, 103].

With recent advances in diffusion model for high-quality image generation, there has also been immense interest in diffusion models for 3D context. Most approaches first train a Vector-Quantized-VAE (VQ-VAE) on a 3D representation such as triplane [10, 64, 82], implicit form [9, 41, 104] and point cloud [33, 102], before employing the diffusion model to the latent space. Direct training on a 3D representation has been less explored. Some recent studies focus on point clouds [52, 62, 108], voxels [107], and neural wavelet coefficients [28, 47]. Our work employs a diffusion model directly on the 3D representation, thereby avoiding information loss associated with the VQ-VAE. Besides formulating our wavelet-tree representation, we propose a scheme to convert it into a format that can be dif-

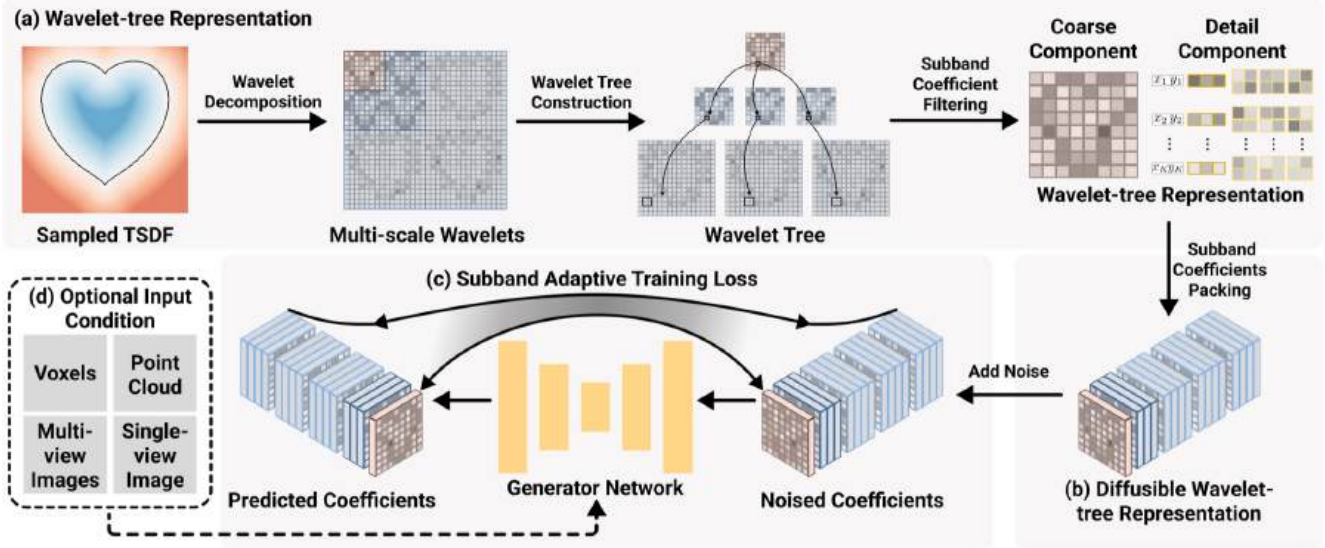


Figure 4. Overview of our generative approach. (a) A shape is first encoded into a truncated signed distance field (TSDF), then decomposed into multi-scale wavelet coefficients in a wavelet-tree structure. We design the *subband coefficient filtering* procedure to exploit the relations among coefficients and extract information-rich coefficients to build our wavelet-tree representation. (b) We propose the *subband coefficient packing* scheme to rearrange our wavelet-tree representation into a regular grid structure of manageable spatial resolution, so that we can adopt a denoising diffusion model to effectively generate the representation. (c) Further, we formulate the *subband adaptive training* strategy to effectively balance the shape information in different subbands and address the detail coefficient sparsity. Hence, we can efficiently train our model on millions of 3D shapes. (d) Our framework can be extended to condition on various modalities.

fusible, or effectively generatable by a diffusion model. Our approach demonstrates great efficiency at high resolutions compared to [107], produces cleaner manifold surfaces than [52, 62, 108] and captures far more details than [28].

Conditional 3D Models. Existing conditional models in 3D can be categorized in two groups. The first group leverages large 2D conditional image generative models, such as Stable Diffusion [74] or Imagen [75], to optimize a 3D scene or object. These methods create 3D shapes and convert them to images using a differentiable renderer, such that the images can be either compared to multiple images or aligned with the distribution of a large text-to-3D generative model. The initial exploration in this area was centered around text-to-3D, as seen in [30, 57, 65]. This approach was later expanded to include images [13, 55, 96] and multi-view images [12, 44, 69, 81]. Recent methods have also incorporated additional conditions such as sketches [58]. Overall, this approach unavoidably requires an expensive optimization, which limits practical applications.

The second group of methods focuses on training a conditional generative model with data that is either paired with a condition or used in a zero-shot manner. Paired conditional generative models explore various conditions such as point cloud [103, 105], image [33, 62, 103, 105], low-resolution voxels [5, 7], sketches [17, 20, 37, 51] and text [33, 62]. More recently, zero-shot methods have gained popularity, with a focus on text [46, 76, 77, 97]

and sketches [78]. In this work, our primary focus is on training a large, paired conditional generative model. This model offers fast generation, as it eliminates the need for scene optimization. Our approach also facilitates the easy incorporation of assorted conditions, *e.g.*, point clouds, low-resolution voxels, and images. Besides, it enables both unconditional applications and zero-shot tasks like shape completion.

3. Overview

Figure 4 provides an overview of our shape-generative framework, designed to create a large-scale 3D generative model capable of efficient training on millions of 3D shapes. The complexity of 3D data makes efficient training at this scale extremely challenging, particularly when considering the need to optimize both the quality of shape generation and the speed of training. Our approach comprises four main components, detailed in Sections 4 to 7.

(i) *Wavelet-tree representation.* We first formulate a compact, efficient and expressive 3D representation to support large-scale shape training. Importantly, we first encode each shape into a high-resolution truncated signed distance field (TSDF) and decompose the TSDF into multi-scale wavelet coefficients. We design a subband coefficient filtering procedure that exploits the relationships among coefficients, allowing us to retain information-rich wavelet

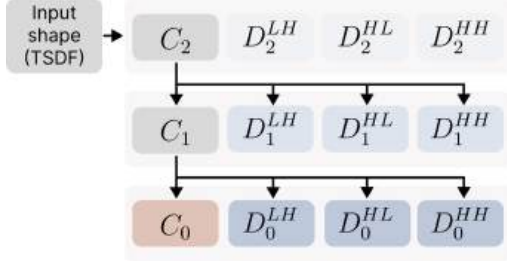


Figure 5. Wavelet decomposition of the input shape, represented as a TSDF, recursively into coarse coefficients C_i and detail coefficients $\{D_i^{LH}, D_i^{HL}, D_i^{HH}\}$. Note that in the 3D case, there will be seven subbands of detail coefficients in each decomposition.

components (both coarse and detail) in our wavelet-tree representation, enabling a faithful yet compact representation of the 3D shape for efficient storage and streaming.

(ii) *Diffusible Wavelet-tree Representation.* Next, we transform the wavelet-tree representation into a format that is more compatible with diffusion models. Though our representation achieves compact shape encoding, its irregular format hinders effective shape learning and generation. This motivates us to design the subband coefficient packing scheme to rearrange the coefficients into a regular grid of a manageable spatial resolution for shape generation.

(iii) *Subband Adaptive Training Strategy.* Further, we exploit methods to train the model on the diffusible wavelet-tree representation. In general, shape information varies across subbands and scales, with detail coefficients being highly sparse yet rich in the shape details. Hence, training with a standard uniform Mean Squared Error (MSE) loss might lead to model collapse or inefficient learning of the details. To address this, we introduce the *subband adaptive training* strategy, which selectively focuses on coefficients in various subbands. This approach allows for an effective balance of shape information over coarse to fine subbands during the training and encourages the model to learn both the structural and detailed aspects of shapes.

(iv) *Extension for Conditional Generation.* Finally, beyond unconditional generation, we extend our method to support conditional generation of shapes, following conditions such as single-/multi-view images, voxels, and point clouds. In essence, we encode the specified conditions into latent vectors and then collectively employ multiple mechanisms to inject these vectors into our generation network.

4. Wavelet-tree Representation

To build our representation, we transform a 3D shape into a truncated signed distance function (TSDF) with a resolution of 256^3 . We decompose the TSDF using a

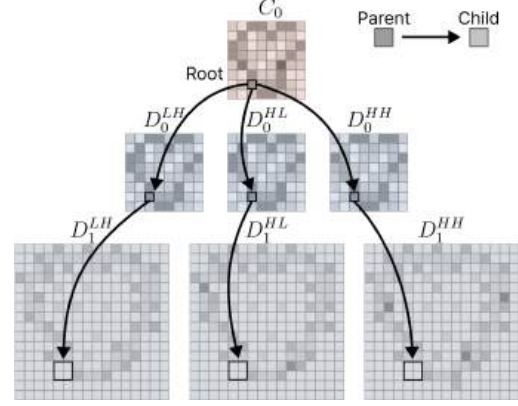


Figure 6. Overview of Parent-child relation. A wavelet tree is formed for each coefficient in C_0 as the root, with a coarser-level coefficient as parent and the finer-level coefficients as children.

wavelet transform¹ into a coarse coefficient C_0 and a set of three detail coefficients $\{D_0, D_1, D_2\}$. The process of obtaining these coefficients involves first transforming TSDF into C_2 and its associated detail coefficients $D_2 = D_2^{LH}, D_2^{HL}, D_2^{HH}$. Then, we decompose C_2 into C_1 and $D_1 = D_1^{LH}, D_1^{HL}, D_1^{HH}$, and subsequently, C_1 is decomposed into C_0 and $D_0 = D_0^{LH}, D_0^{HL}, D_0^{HH}$. This process is depicted in Figure 5. For simplicity, we present our method using 2D illustrations, yet the actual computation is performed in 3D with seven *subband volumes* (instead of three *subband images*, in the 2D case) of detail coefficients in each decomposition. It is important to note that the detail coefficients contain high-frequency information. Furthermore, this representation is lossless and can be bijectively converted to a TSDF through inverse wavelet transforms.

Wavelet-tree and Coefficient relation. Building upon the neural wavelet representation as proposed in [28], we propose to exploit the relationships between wavelet coefficients for our 3D representation. Generally, each coarse coefficient in C_0 , referred to as a *parent*, and its associated detail coefficients in D_0 , known as *children*, reconstruct the corresponding coefficients in C_1 through an inverse wavelet transform. This *parent-child relation* relationship extends between D_0 and D_1 , and so forth, as shown by the arrows leading from D_0 to D_1 in Figure 6. Additionally, coefficients sharing the same parent are termed *siblings*. By aggregating all descendants of a coefficient in C_0 , we can construct a *wavelet coefficient tree* or simply a wavelet tree, with a coefficient in C_0 serving as its root. This concept is further illustrated in Figure 6.

Observations. We have identified four notable observations regarding the wavelet coefficients:

¹Following the approach in [28], we use biorthogonal wavelets with 6 and 8 moments.

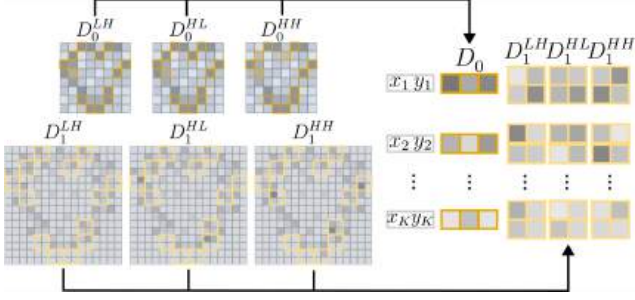


Figure 7. The detail component part of our representation. We extract and pack informative coefficients from D_0 and D_1 , indicated in yellow boxes, along with their spatial locations to form our representation’s detail component.

- (i) If a coefficient’s magnitude is smaller than a threshold (say, $1/32$ of the largest coefficient in a subband), its children will likely have small magnitudes. Small magnitude means low contribution to the shape, so these coefficients have little impact on the shape. We empirically studied this observation in the D_0 subbands of 1,000 random shapes and found that more than 96.1% of the coefficients satisfy this hypothesis.
- (ii) The values of sibling coefficients are positively correlated. We evaluated the correlation coefficients between all pairs of sibling coefficients in 1,000 random shapes and found a positive correlation value of 0.35.
- (iii) Coefficients in C_0 are mostly non-zeros, with a mean magnitude of 2.2, while the mean magnitude of detail coefficients in D_0 are much closer to zero, implying that C_0 contains most of the shape information.
- (iv) Most coefficients in D_2 are insignificant. By empirically setting them to zeros in inverse wavelet transforms, we can reconstruct the TSDFs faithfully for 1,000 random shapes with 99.64% IoU accuracy.

Subband Coefficient Filtering. Based on the observations, we design the subband coefficient filtering procedure to locate and pack information-rich coefficients when building our representation. First, we keep all the coefficients in C_0 , take them as the *coarse component* in our representation, and exclude all the coefficients in D_2 , following observations (iii) and (iv). Second, C_0 alone is insufficient to capture the details; compare Figure 3 (d) vs. (e). We need the coefficients in D_0 and D_1 . However, simply including all coefficients in D_0 and D_1 will lead to a bulky representation. Hence, we aim for a compact representation that can retain details by following observations (i) and (ii) to exploit the coefficient relations in D_0 and D_1 .

Procedure-wise, since the subbands of D_0 share the same resolution and are positively correlated, as analyzed in observation (iii), we collectively examine the coefficient locations in all subbands of D_0 together. For each coefficient

location, we examine the sibling coefficients in D_0^{LH} , D_0^{HL} , and D_0^{HH} , selecting the one with the largest magnitude. We consider its magnitude value as the measure of *information* for that coefficient location. Next, we filter out the top K coefficient locations with the highest information content (refer to Figure 7 on the left) and store their location coordinates and associated coefficient values in D_0 , along with their children’s coefficient values in D_1 . This forms the *detail component* in our wavelet-tree representation, as illustrated in Figure 7 on the right. Together with the coarse component, i.e., C_0 , we construct our wavelet-tree representation. Despite excluding all the coefficients in D_2 and selecting the top K coefficients in D_0 , our representation can effectively achieve an impressive mean IOU of 99.56%.

To efficiently process millions of 3D shapes, we utilize our wavelet-tree representation for encoding each shape, subsequently storing the results in the cloud. This constitutes a one-time preprocessing step. This aspect is particularly crucial for large-scale training, as it results in a 44.5% reduction in both data streaming and loading, a significant improvement over directly using the 256^3 SDF, as pointed out in the introduction. Moreover, we deliberately avoid using more sophisticated compression techniques due to the decompression overhead they incur, which can significantly slow down the model training.

5. Diffusible Wavelet-tree Representation

Next, we develop a representation that can be effectively trained and generated by a diffusion-based generative model. This diffusion model is based on the DDPM framework [24], which formulates the generative process as a Markov chain. This chain comprises two key processes: (i) a forward process, which incrementally introduces noise into a data sample x_0 over T time steps, eventually transforming it into a unit Gaussian distribution, denoted as $p(x_T) \sim N(0, I)$; and (ii) a reverse process, which is characterized by a generator network θ tasked to progressively remove noise from a noisy sample. In our approach, the generator network is designed to predict the original diffusion target x_0 directly from the noisy sample x_t , expressed as $f_\theta(x_t, t) \simeq x_0$. To achieve this, we employ a mean-squares loss objective.

Challenges. Our wavelet-tree representation, while being compact and efficient for data streaming, it encounters specific challenges during training. This representation consists of a coarse component, C_0 , structured as a grid, and a detail component containing three irregular arrays. The detail component is derived from D_0 and D_1 , as illustrated in Figure 7 (right). A straightforward approach is to directly treat this representation as the diffusion target and predict the coarse and detail components using a two-branch network. However, it is hard to accurately predict the detail co-

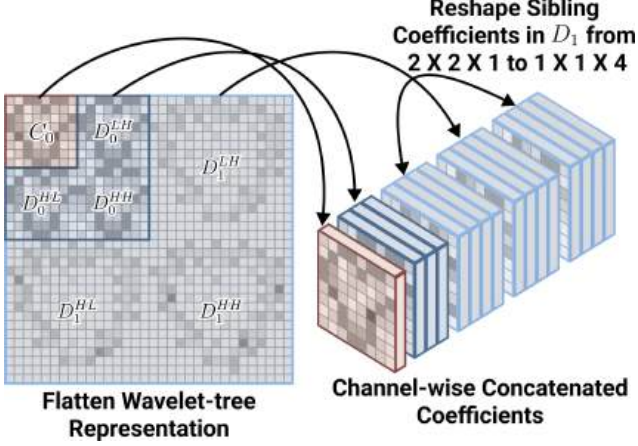


Figure 8. Diffusible wavelet representation. First, we unpack and flatten the coefficients in our wavelet-tree representation (left). Following observation (iii), we channel-wise concatenate sibling coefficients to reduce the spatial resolution (right). Here we concatenate each coefficient in C_0 with its three children in D_0 and the reshaped descendants in D_1 (each of size $1 \times 1 \times 4$).

efficient locations while balancing multiple objectives. We empirically observed that this approach struggles with convergence and leads to the collapse of model training.

Another approach we tried is to flatten the extracted coefficients in our representation to avoid the irregularities in the detail component. As Figure 8 (left) shows, we first arrange the coarse component C_0 at top left, then pack D_0 and D_1 successively around C_0 by arranging the extracted detail coefficients at their respective locations in each subband while leaving the remaining locations as zeros. In this way, the input representation becomes a regular 2D grid for the DDPM to model. However, this representation is spatially very large. The current U-Net architecture, widely adopted by existing diffusion models, creates GPU memory-intensive feature maps, which can lead to out-of-memory issues and result in low computational intensity, thus leading to poor utilization of the accelerator [26]. Consequently, model training remains intractable.

Subband Coefficients Packing. To address these challenges, we draw inspiration from recent work on efficient 2D image generation [26]. Our motivation is further bolstered by observation (iii), which highlights the relationship between coefficients. This insight enables us to effectively pack sibling subband coefficients, exhibiting similar structures (as illustrated in D_0 and D_1 of Figure 6), along with their corresponding children. These children are reshaped into a $1 \times 1 \times 4$ format, as demonstrated in Figure 8 (right), and integrated into the channel dimension. In this approach, the resulting representation (the diffusion model target) adopts a grid structure with reduced spatial resolution but an increased number of channels. Hence,

this allows us to circumvent the use of memory-intensive feature maps, avoiding out-of-memory issues and enabling more efficient computation when training on millions of 3D shapes. Using this strategy in our 3D representation can approximately lead to a cubic-order speedup and a significant reduction in GPU memory usage, estimated to be around 64x, when applied to the same network architecture.

6. Subband Adaptive Training Strategy

In this step, our goal is to effectively train our diffusion model on this diffusible wavelet-tree representation. A key challenge for our model is ensuring that it can proficiently generate both the coarse and detailed components of the representation. An obvious choice for this purpose would be to employ a standard mean squared error (MSE) loss on the coefficient set X :

$$L_{\text{MSE}}(X) := \frac{1}{|X|} \sum_{x_0} \|f_{\theta}(x_t, t) - x_0\|^2, x_0 \in X, \quad (1)$$

where x_t is the noised coefficient of x_0 at time step t . Empirically, we observed that naively applying MSE to all coefficients results in significant quality degradation during training, as demonstrated in our ablation studies. We attribute this to two primary factors. Firstly, there is an imbalance in the number of coefficients across different scales. In the 2D case, the ratio of $|C_0| : |D_0| : |D_1|$ is $1 : 3 : 12$, and in the 3D case, it is $1 : 7 : 56$. Consequently, using a uniform MSE loss tends to disproportionately emphasize the fine-detail coefficients, even though the core shape information is more densely represented in C_0 than in D_0 and D_1 . Secondly, the majority of the detail coefficients are close to zeros, with only a few having high magnitude or *information* which contain maximum high-frequency information about the shape. Therefore, uniformly sampling a loss across these coefficients can result in a sub-optimal training mechanism due to the imbalance in the number of high-magnitude detail coefficients.

An initial approach to tackle the issue involves defining three separate MSE losses for C_0 , D_0 , and D_1 , then combining these losses with equal weights. However, this approach still treats the losses on detail coefficients uniformly, without resolving the imbalance issue due to sparsity in the detail subbands. This oversight is empirically evidenced by the subpar performance observed in our ablation study.

Subband Adaptive Training. To address these issues, we develop a subband adaptive training strategy. This approach is specifically designed to focus more effectively on the high magnitude detail coefficients while still maintaining a balanced consideration for the other remaining detail coefficients. This ensures that they are not completely overlooked. Specifically, for each subband in D_0 , say D_0^{LH} , we first locate the coefficient of the largest magnitude in D_0^{LH} .

Denoting v as its magnitude value, we then identify all coefficients in D_0^{LH} with magnitude larger than $v/32$; we regard these coefficients as important, and record their spatial locations into coordinate set P_0^{LH} . Similarly, we can obtain coordinate sets P_0^{HL} for D_0^{HL} and P_0^{HH} for D_0^{HH} . As sibling coefficients are positively-correlated, we union the three coordinate sets into coordinate set P_0 , which records the spatial locations of the important detail coefficients.

On the other hand, we define P'_0 as the coordinate set complement to P_0 in the spatial domain of the D_0 subband. Using P_0 and P'_0 , we can then formulate the training loss as

$$L_{\text{MSE}}(C_0) + \frac{1}{2} \left[\sum_D L_{\text{MSE}}(D[P_0]) + \sum_D L_{\text{MSE}}(R(D[P'_0])) \right], \quad (2)$$

where D is a subband in $\{D_0, D_1\}$; $D[P]$ denotes D 's coefficients at locations in P ; and R is a function that randomly picks some of the coefficients in $D[P'_0]$. Importantly, the number of coefficients picked by R is $|P_0|$, such that we can balance the last two terms in our loss with the same number of coefficients. Note that using random sampling can effectively regularize the less important small coefficients while not completely ignoring them in the training. Also, P_0 includes far less coefficients than the whole domain of D_0 (only $\sim 7.4\%$), so the model training can effectively focus on the crucial information while attending to the details.

Efficient Loss Computation. To calculate the losses, we may store P_0 and P'_0 as irregularly-sized coordinate sets and employ slicing to process the generation target and network prediction. Doing so is, however, highly inefficient, as it requires distinct computations for different data samples and prevents us from leveraging code compilation features for training speedup.

To improve the model training efficiency, we propose to use a fixed-size binary mask to represent the coordinate set, where a value of one indicates the associated location as selected. The MSE loss can then be efficiently calculated by masking both the generation target and network prediction. This approach eliminates the need for irregular operations and allows for efficient use of code compilation in PyTorch to accelerate the training.

7. Extension for Conditional Generation

. Our framework is versatile and can be extended beyond unconditional generation to accommodate conditional generation across various modalities. To achieve this, we adopt different encoder for each modality that transforms a given condition into a sequence of latent vectors. Subsequently, these vectors are injected into the generator using multiple conditioning mechanisms. We also use a classifier-free guidance mechanism [23], which has empirically demonstrated greater effectiveness in conditional settings.

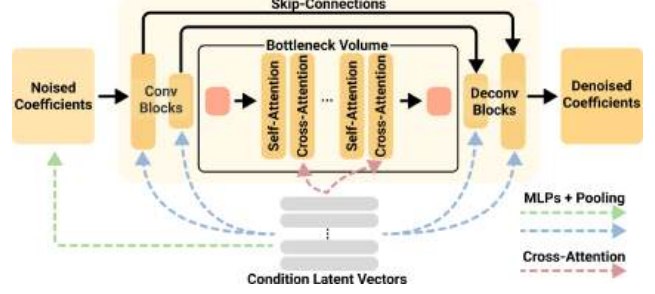


Figure 9. Our generator network progressively downsamples input coefficients to a bottleneck feature volume (middle). This volume goes through attention layers and deconvolution for upsampling to predict the denoised coefficients. If the condition latent vectors are available, we simultaneously transform these vectors and adopt them at three locations in our architecture: (i) concatenating with the input noised coefficients (the green arrow); (ii) conditioning the convolution and deconvolution blocks (the blue arrows); and (iii) cross-attention with the bottleneck volume (the red arrows).

Condition Latent Vectors. We deliberately convert all input conditions into a sequence of latent vectors, which we call *condition latent vectors*, to preserve the generality of our conditioning framework. This approach eliminates the need to devise new specific condition mechanisms to diffusion model for each modality, thereby enabling our framework to function seamlessly across various modalities. Our encoder for different modality are described below:

- (i) *Single-view image.* Given a rendered image of a 3D model, we utilize the pre-trained CLIP L-14 image encoder [70] to process the image. The latent vectors extracted from just before the pooling layer of this encoder are then used as the conditional latent vectors.
- (ii) *Multi-view images.* We are provided with four images of a 3D model, each rendered from one of 55 pre-defined camera poses (selected randomly). To generate the conditional latent vectors, we first use the CLIP L-14 image encoder to process each rendered image individually to produce an image latent vector. Considering the camera poses, we maintain 55 trainable camera latent vectors, each corresponding to one camera pose and matching the dimensionality of the latent vectors encoded by the CLIP image encoder. For each encoded image latent vector, we retrieve the corresponding trainable camera latent vector based on the camera pose of the image. This camera vector is then added to each image latent vector in the sequence in an element-wise fashion. Finally, the four processed sequences of latent vectors are concatenated to form the conditional latent vectors.
- (iii) *3D point cloud.* We utilize three Multi-Layer Perceptron (MLP) layers to first transform the given point cloud into feature vectors like PointNet [67]. These

vectors are then aggregated using the PMA block form the Set Transformer layer [39], resulting in sequence of latent vectors that serve as the condition.

- (iv) *Voxels*. We utilize two 3D convolution layers to progressively downsample the input 3D voxels into a 3D feature volume. This volume is subsequently flattened to form the desired conditional latent vectors.

Network Architecture. Figure 9 illustrates the network architecture of our generator. The main branch, highlighted by yellow boxes, adopts a U-ViT architecture [26]. The network uses multiple ResNet convolution layers for down-sampling our noised coefficients into a feature bottleneck volume, as shown in the middle part of Figure 9. Following this step, we apply a series of attention layers to the volume. The volume is then upsampled using various deconvolution layers to produce the denoised coefficients. A key feature of our architecture is the inclusion of learnable skip-connections between the convolution and deconvolution blocks, which have been found to enhance stability and facilitate more effective information sharing [27].

Moreover, when condition latent vectors are available, we integrate them into our generation network at three distinct locations in the U-ViT architecture, as depicted in the bottom part of Figure 9. Initially, these latent vectors are processed through MLP layers and a pooling layer to yield a single latent vector (highlighted by the green arrow in the left section of Figure 9). This vector is subsequently concatenated as additional channels of the input noise coefficients. Second, following a similar process, we convert condition latent vectors to another latent vector. However, this vector is utilized to condition the convolution and deconvolution layers via modulating the affine parameters of group normalization layers [14]. This integration is represented by the blue arrows in Figure 9. Lastly, to condition the bottleneck volume, an additional positional encoding is applied to the condition latent vectors in an element-wise fashion. These vectors are then used in a cross-attention operation with the bottleneck volume, as indicated by the red arrows in Figure 9.

8. Results

In this section, we begin by presenting the experimental setup, followed by both quantitative and qualitative results obtained using various input conditions. Further, we show-case that our generative model is adaptable to shape completion tasks without additional training. Last, we present comprehensive analysis and ablations on our framework.

8.1. Experimental Setup

Dataset. We compile a new very large-scale dataset, consisting of more than 10 million 3D shapes, from 18 existing publicly-available sub-datasets: ModelNet [89],

ShapeNet [3], SMLP [48], Thingi10K [109], SMAL [110], COMA [73], House3D [94], ABC [36], Fusion 360 [91], 3D-FUTURE [16], BuildingNet [80], DeformingThings4D [42], FG3D [45], Toys4K [85], ABO [11], Infinigen [71], Objaverse [12], and two subsets of ObjaverseXL [12] (Thingiverse and GitHub). Among the sub-datasets, some contain specific object classes, *e.g.*, CAD models (ABC and Fusion 360), furniture (ShapeNet, 3D-FUTURE, ModelNet, FG3D, and ABO), humans (SMLP and DeformingThings4D), animals (SMAL and Infinigen), plants (Infinigen), faces (COMA), houses (BuildingNet and House3D), etc. Beyond these, the Objaverse and ObjaverseXL datasets encompass generic objects collected from the internet, thereby not only covering the aforementioned categories but also offering a more diverse range of objects. For the data split, we randomly divided each sub-dataset into two parts: a training set consisting of 98% of the shapes and a testing set comprising the remaining 2%. The final train and test sets were then compiled by combining the respective train and test sets from each sub-dataset.

For each shape in our dataset, we generate a TSDF and its wavelet-tree representation for model training. On the other hand, we prepare various additional inputs for the conditional generation tasks. For image inputs, we randomly sampled 55 pre-defined camera poses and rendered 55 images for each object according to these poses, using the scripts provided by [33]. For voxel inputs, we prepared two sets of voxels at different resolutions (16^3 and 32^3) for each 3D object and trained separate models for each resolution. Lastly, we randomly sampled 25,000 points on the surface of each 3D shape to generate the point cloud input.

Training Details. We train our shape model using the Adam Optimizer [34] with a learning rate of $1e-4$ and a batch size of 96. To stabilize the training, we employ an exponential moving average with a decay rate of 0.9999 in the model updates, in line with existing 2D large-scale diffusion models [74]. Our model is trained on $48 \times A10G$ with 2M-4M iterations, depending on the input condition.

Evaluation Dataset. For qualitative evaluation, we provide visual results based on the inputs in the unseen test set of our compiled large-scale dataset. For quantitative evaluation, we prepare two evaluation sets for metric computation. In particular, we randomly select 50 shapes from the test set of each sub-dataset to form the first evaluation set for computing the metrics. We refer to this dataset as “Our Val” dataset throughout the remainder of the paper. We also utilize the Google Scanned Objects (GSO) dataset to create an additional evaluation set to evaluate the cross-domain generalization capability for our method, noting that our model has not been trained on this dataset. Please note that while a subset of the Google Scanned Objects (GSO) dataset has been used as an evaluation set in One-2345 [43], we have included all objects from this dataset in our study to ensure

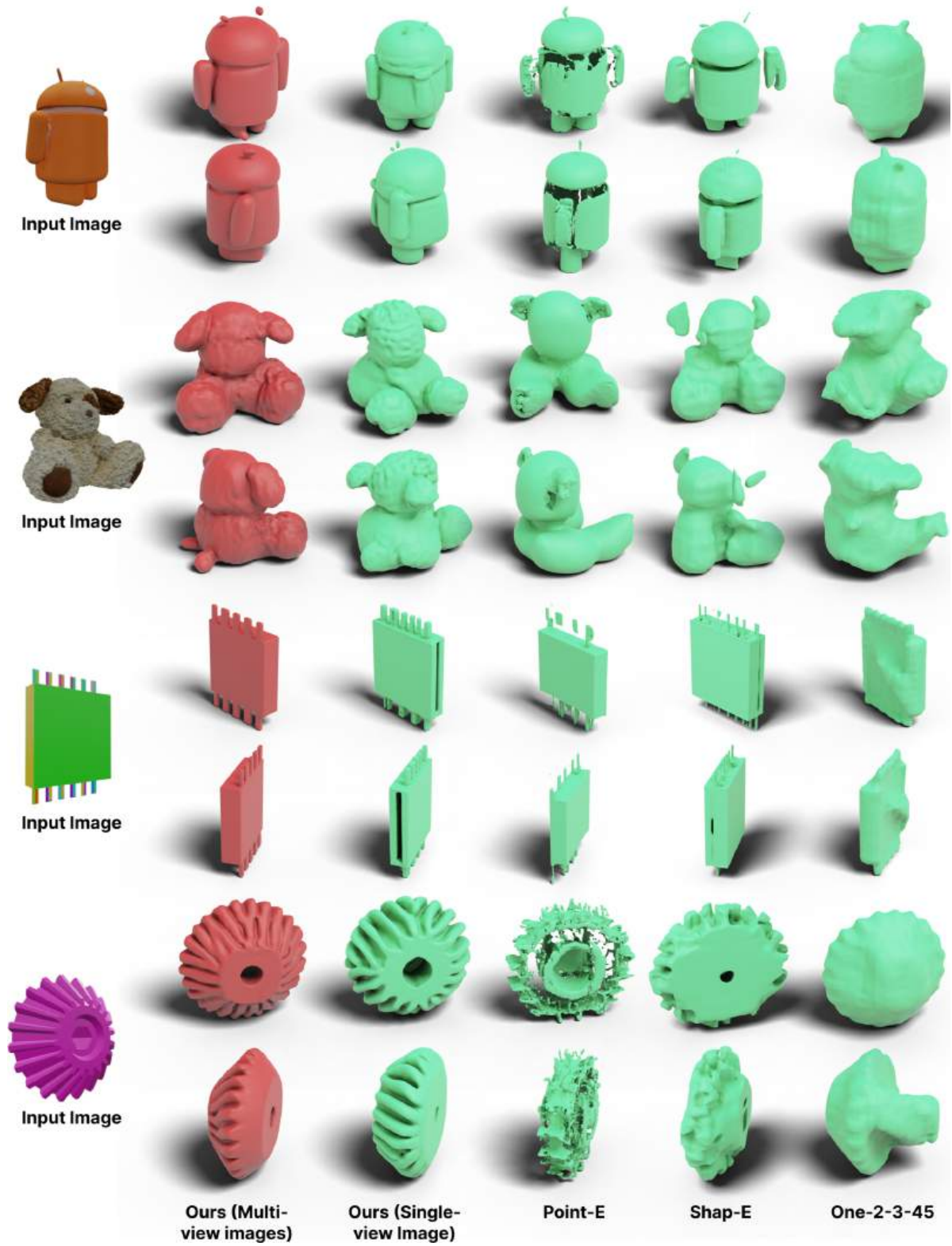


Figure 10. Visual comparisons for the Image-to-3D generation task reveal that our method outperforms three major generative models: Point-E [62], Shap-E [33], and One-2-3-45 [43]. Our single-view model generates more accurate shapes compared to these baselines, and the multi-view model further enhances shape fidelity with additional view information.

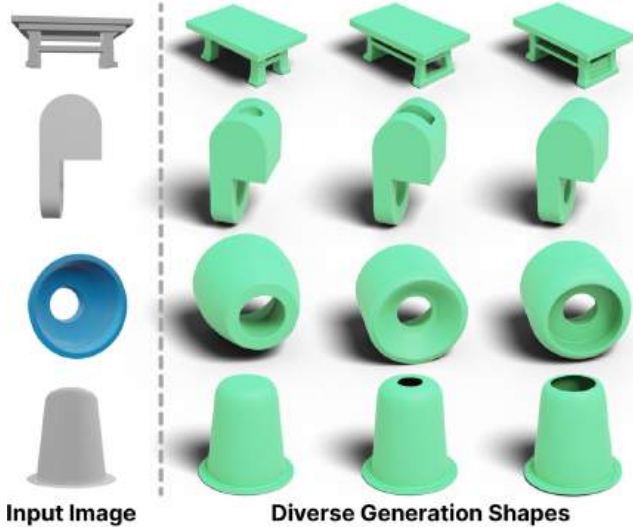


Figure 11. Our model demonstrates the capability to generate varied results from a single input image, accurately resembling the visible portions while offering diversity in unseen areas.

Table 3. Quantitative evaluation of the Image-to-3D task shows that our single-view model excels the baselines, achieving the highest IoU and lowest LFD metrics. Incorporating additional information, our multi-view model further enhances performance.

<i>Method</i>	GSO Dataset		Our Val Dataset	
	LFD ↓	IoU ↑	LFD ↓	IoU ↑
Point-E [62]	5018.73	0.1948	6181.97	0.2154
Shap-E [33]	3824.48	0.3488	4858.92	0.2656
One-2-3-45 [43]	4397.18	0.4159	5094.11	0.2900
Ours (Single view)	3406.61	0.5004	4071.33	0.4285
Ours (Multi view)	1890.85	0.7460	2217.25	0.6707

a more comprehensive evaluation.

Evaluation Metrics. In the conditional task, we evaluate performance by comparing the similarity between the generated shape and the associated ground-truth shape using two metrics: (i) Intersection over Union (IoU), which measures the volume ratio of the intersection to the union between the voxelized volumes; and (ii) Light Field Distance (LFD) [4], which assesses the similarity between two sets of images rendered from various viewpoints.

For the unconditional task, we adopt the method described in [104] to evaluate the generation performance using the Frechet Inception Distance (FID) [22]. In particular, we render an image for each shape from Our Val set. Following this, we apply the same rendering process to a generated set of shapes of equivalent size. We then compute a feature vector for each rendered image and assess the difference in the distribution of these feature vectors between the two sets as the final metric.

8.2. Quantitative Comparison with Other Large Generative Models

In this experiment, we contrast our method with other large image-to-3D generative models. Our analysis encompasses two distinct model settings: single-view and multi-view. The single-view model uses a solitary image, which serves as an input for our wavelet generative model. In cases where multiple images are accessible, our multi-view model comes into play. This model uses four images along with the camera parameter as the condition.

We present the quantitative results in Table 3 and the qualitative comparison results in Figure 10. As shown in Table 3, our single-view model significantly outperforms all baseline models by a considerable margin in both the IoU and LFD metrics. It is noteworthy that LFD, being a rotation-insensitive metric, indicates that our results do not depend on the alignment between the generated shapes and the ground-truth shapes. Additionally, Figure 10 reveals that our method captures not only global structures but also fine local details, as well as more intricate geometric patterns, compared to the baselines. This is particularly evident in rows 7-8 of Figure 10, showing that our method is able to more accurately capture the geometric patterns of lines than the other baselines. Furthermore, rows 3-4 showcase how our model effectively captures the rough coat pattern of the shape, further demonstrating its superiority. Finally, we would like to highlight that the geometry produced by our method features clean and smooth surfaces, in contrast to those generated by other baseline methods.

On the other hand, when provided with additional three views, our model exhibits a substantial improvement in results. Both LFD and IoU metrics, as indicated in Table 3, show that the generative model captures a greater extent of the global shape. This improvement is expected, since the network can access more information in the form of multiple views; however, four images still constitute a limited set of views for completely reconstructing a shape. Moreover, multiple views aid in better capturing local geometric details, as demonstrated in Figure 10 rows 3-4 and 7-8. Also, it is worthy to mention recent advancements, such as Zero123 [44] and Zero123-XL [12], which have the capability to generate multiple views from a single view. This advancement potentially allows our multi-view model to operate effectively with only a single view available since a single-view can be converted to a multi-view using Zero123 or Zero123-XL, after which our model can be applied. Yet, exploring this possibility remains a topic for future research.

We attribute this success to two major factors. Firstly, we posit that our wavelet-tree representation is almost lossless, as discussed in Table 1, compared to Point-E and Shape-E. This characteristic likely makes the upper bound of our representation much easier to capture using a diffusion model. Moreover, our adaptive training scheme enables our net-

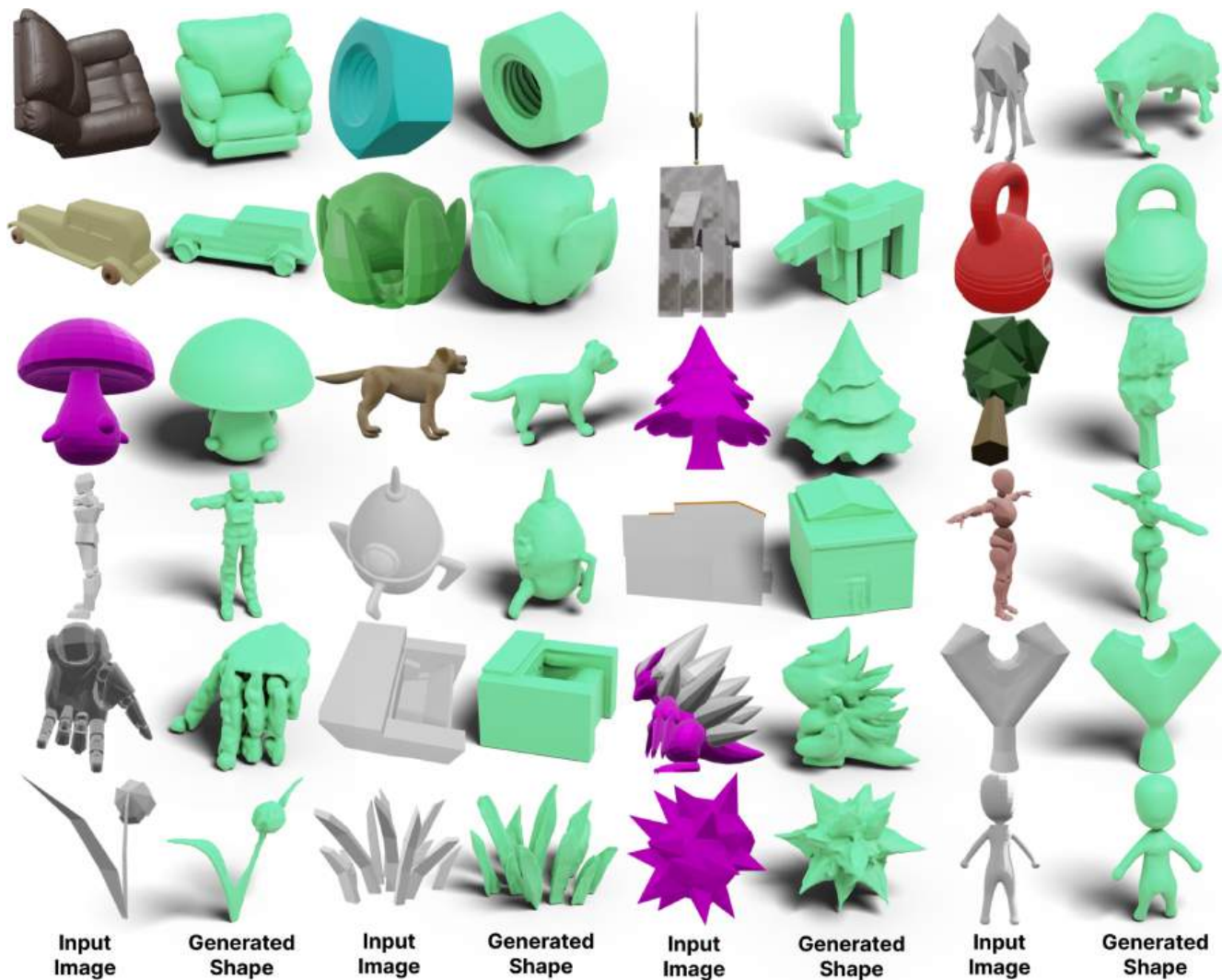


Figure 12. Our single-view conditioned generation model yields a wide variety of shapes. Our model adeptly generates both CAD objects like screws, chairs, and cars, as well as organic forms such as humans, animals, and plants.

work to capture more local details and geometric patterns, by efficiently learning to model the information-rich coefficients in the detail coefficient subbands.

8.3. Image-to-3D Generation

In this section, we present additional qualitative results on single-view conditioning from our val set. As depicted in Figure 12, it is evident that our method can generate objects from a variety of categories. This includes CAD objects, such as the screw shown in row 1, furniture like the chair in row 5, and buildings exemplified by the house in row 4. Additionally, our method effectively represents organic shapes, including humans (as seen in rows 4 and 6), animals (with the dog in row 3), and plants (illustrated by the Christmas tree and mushroom in row 3).

Our model, being a generative one, can produce multiple

variations for a given condition, as demonstrated in Figure 11 using single-view images. In addition to the visible regions that can be faithfully reconstructed by our approach, it further imaginatively reconstructs the parts that are invisible. For instance, in Figure 11, rows 2 and 4, the top of the CAD object is invisible, leading the model to imagine it. This trade-off between adhering to visible parts and creatively interpreting invisible ones could be further explored by adjusting the classifier-free guidance weight. We leave this aspect for future work.

We also present additional results for the multi-view approach in Figure 13. Again, it demonstrates that our method is able to generate objects across diverse categories. Moreover, there is a noticeable alignment of the objects with the input images, which is more pronounced compared to the single-view approach. Our method is also capable of gen-

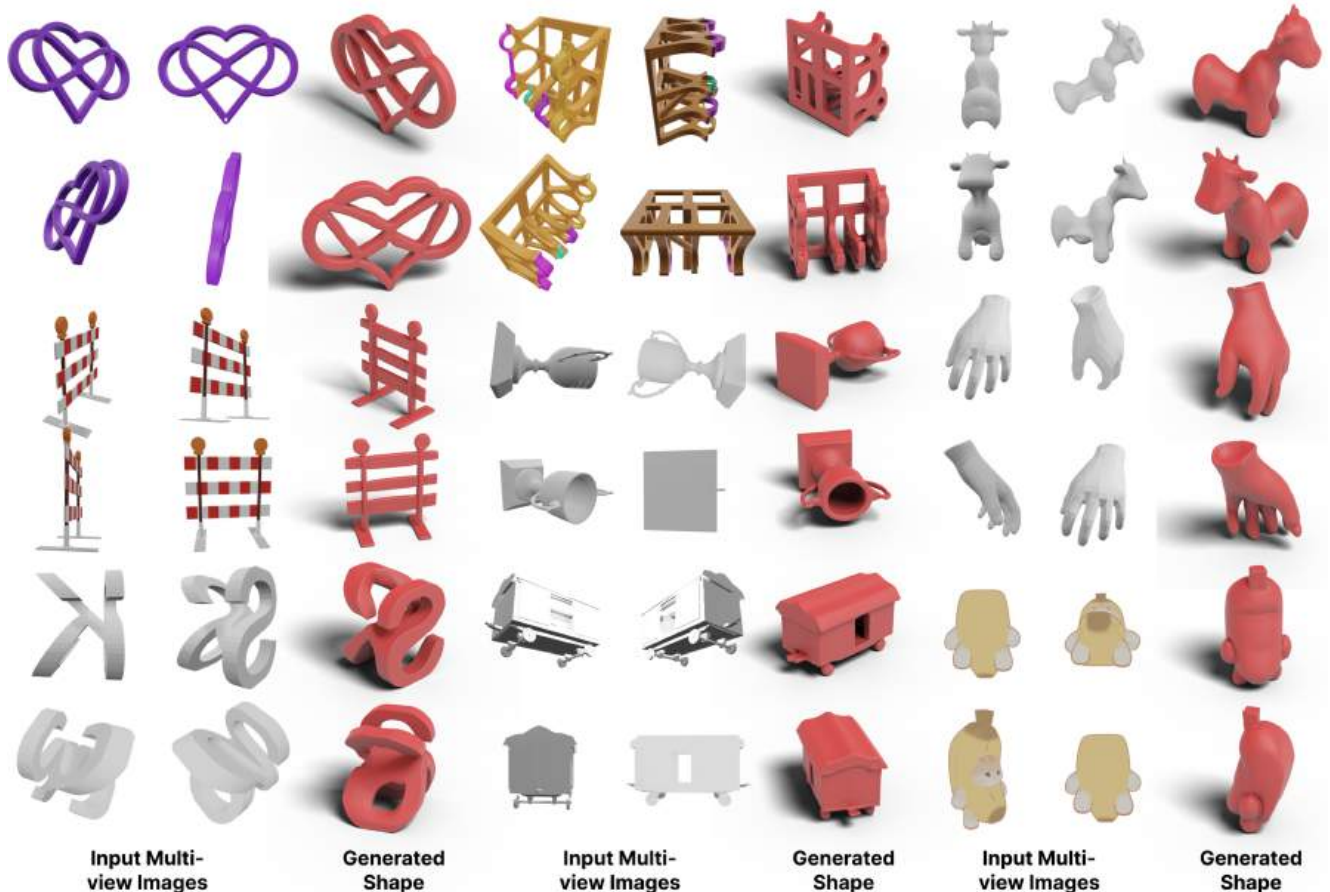


Figure 13. Our multi-view conditioned generation model can utilize multi-view information to create diverse and coherent shapes with complex topologies, exemplified by the CAD objects in the first two rows.

Table 4. The quantitative evaluation reveals that our model’s performance is not significantly impacted by the number of points of inputs. Even with inputs of 5000 points, it manages to deliver reasonable reconstructions, though trained on 25000-point inputs.

Metrics	Number of Points			
	2500	5000	10000	25000
LFD ↓	1857.84	1472.02	1397.39	1368.90
IoU ↑	0.7595	0.8338	0.8493	0.8535

erating objects with highly complicated topologies and intricate geometries, as demonstrated by the CAD examples (see the second object in rows 1-2 and the first object in rows 5-6).

8.4. Point-to-3D Generation

In this experiment, our objective is to take a point cloud as input and produce a TSDF following its geometry. Our encoder, comprising a PointNet [67] and a PMA block from the Set Transformer [39], is adept at handling a variable

number of points. This versatility allows us to train the model with 25,000 points, while also giving us the flexibility to input an arbitrary number of points during testing.

We conduct an ablation study to assess how the quality of generation is influenced by different sets of points, as detailed in Table 4. Our findings reveal that an increase in the number of points leads to improved IoU results on our val set. Notably, even with sparse point clouds with as few as 5,000 points, our model achieves a reasonable IoU.

This analysis is also visually represented illustrated in Figure 14. Here, we observe that certain details are lost when a lower number of points are used, as evident in row 2. However, it’s worth mentioning that, in general, our method performs well even with fewer points. We also present additional visual results in Figure 15. These results demonstrate that our method performs consistently well across various types of object and showcasing robustness predominantly to the number of points.

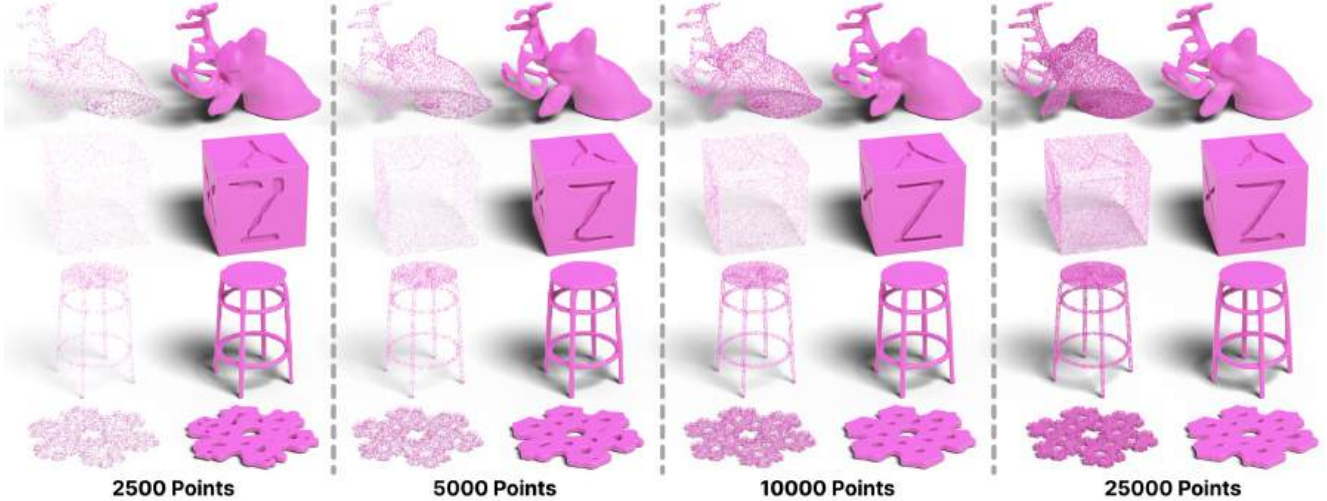


Figure 14. Visual comparisons, based on the number of input points, highlight our model’s ability to robustly generate thin structures, like the deer horn or the chair leg, with a reasonable number of points (≥ 5000).

Table 5. Our method is quantitatively compared with traditional voxel upsampling techniques, specifically nearest neighbour upsampling and trilinear interpolation, followed by marching cubes [49] for mesh extraction. Our generative model significantly surpasses these two baselines in both Light Field Distance (LFD) and Intersection Over Union (IoU) metrics.

Setting	Methods	LFD ↓	IoU ↑
Voxel (16^3)	Ours	2266.41	0.687
	Nearest	6408.82	0.2331
	Trilinear	6132.99	0.2373
Voxel (32^3)	Ours	1580.98	0.7942
	Nearest	3970.49	0.4677
	Trilinear	3682.83	0.4719

8.5. Voxel-to-3D Generation

Next, we explore the use of low-resolution voxels as input for our model, which outputs a Signed Distance Function following its geometry. We trained two distinct models on different voxel resolutions: 16^3 and 32^3 . Both models employ the same encoder, which utilizes two 3D convolutional blocks to downsample the input into a conditional latent vector.

The qualitative results for both resolutions are displayed in Figure 16. From this analysis, we observe three main points. First, our method successfully creates smooth and clean surfaces. Second, despite the ambiguity present in some examples for both voxel 16^3 (as seen in the human examples in row 3) and voxel 32^3 (as seen in the crown examples in row 5), our method produces plausible shapes. Finally, our method also performs well with disjoint objects (as demonstrated in the human examples in row 3) and

scenes (as shown in the room examples in row 1 and row 7).

We further compare our method with traditional approaches for converting low-resolution voxels to meshes. For the baselines, we first employ interpolation techniques such as nearest neighbor and trilinear interpolation, followed by the use of marching cubes [49] to derive the meshes. Importantly, our approach is the first large-scale generative model to tackle this task. The quantitative and qualitative results of this comparison are presented in Table 5 and Figure 17. It is evident that, among the baseline methods, trilinear interpolation outperforms nearest neighbor, which is intuitively reasonable. Our method easily surpasses both of these traditional methods in terms of both IoU and LFD metrics.

8.6. 3D Shape Completion

Furthermore, our trained unconditional generative model can be employed for completion tasks in a zero-shot fashion. In this context, the objective is to generate multiple variations of a partial input shape. Given a shape, we first identify a region, consisting of a set of vertices on the mesh, that we aim to regenerate and subsequently discard it. The remaining part of the mesh is retained as the partial input as shown in Figure 18 (leftmost column). Subsequently, this partial shape is transformed into a TSDF, which is then converted into our diffusible wavelet representation with a grid structure. Based on the earlier selected region for regeneration, we construct a binary mask that identifies missing regions to complete. Utilizing the selection mask, we adopt the approach described in [50] to regenerate the wavelet coefficients in these masked regions using our trained unconditional model.

Figure 18 shows visual examples of our completion ap-

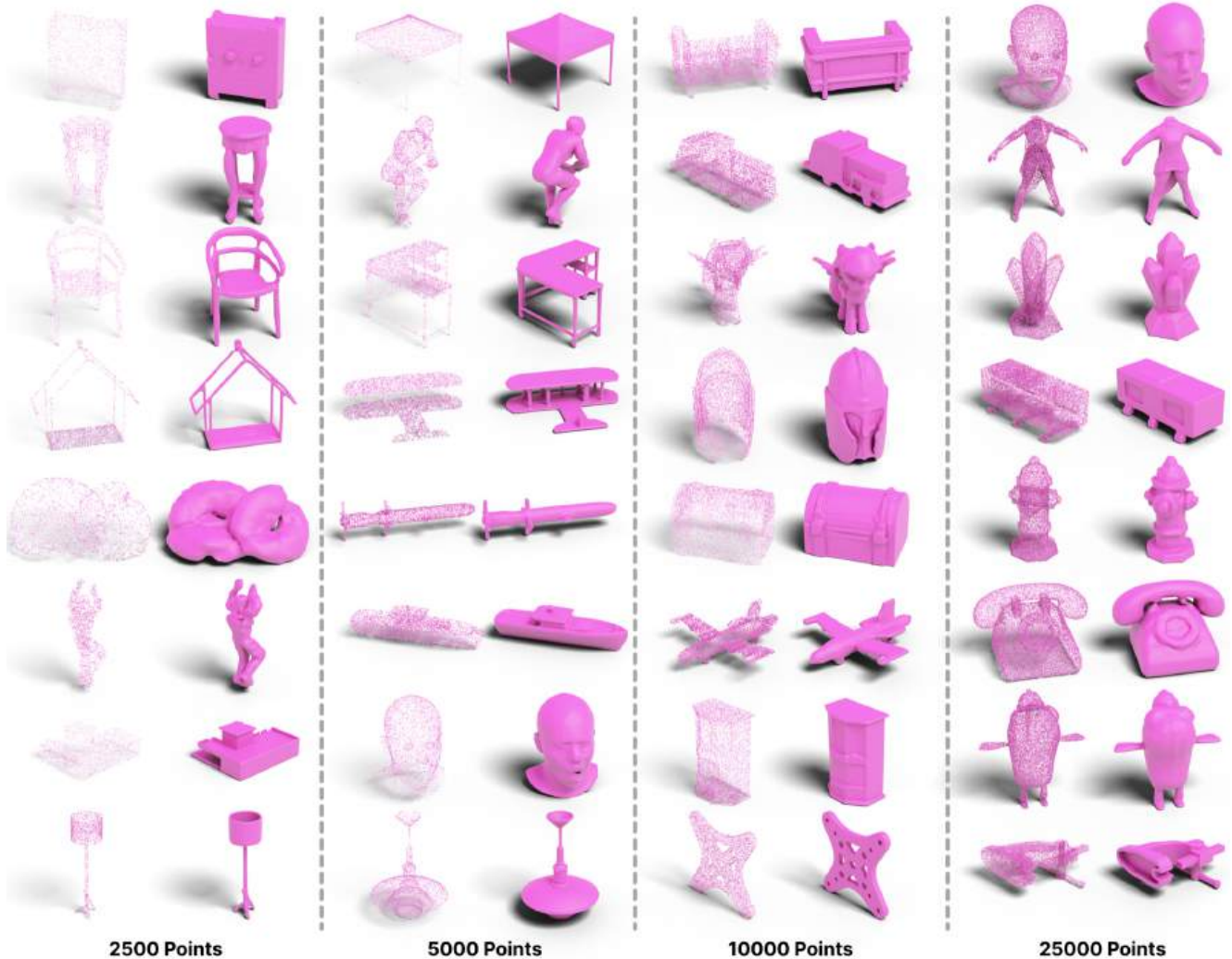


Figure 15. Our point cloud conditioned generation results demonstrate that our model is capable of producing shapes with intricate geometric details while preserving the complex topologies of the input point clouds.

proach. It is evident that, in general, our unconditional model can generate semantically meaningful parts in the masked regions, as illustrated by the animal heads produced in the third row and the varied chair backs in the first row. Additionally, these generated parts exhibit coherent connections to the given input of the partial shape, exemplified by the body region of the animal in the fifth row. It is also noteworthy that our model can produce multiple variations for the same input partial shape, indicating the diverse distribution captured in our generative model.

8.7. Ablation Studies

Classifier-free Guidance. As previously mentioned, we employ classifier-free guidance, as detailed in [23], to enhance the quality of conditioned samples. A crucial hyperparameter in this classifier-free guidance, during inference,

Table 6. We quantitatively analyse the performance of our conditional generation models on different guidance weights.

Model	Metrics	Guidance Weight (w)				
		1	2	3	4	5
Single-view	LFD ↓	4395.15	4071.33	4121.14	4192.30	4295.28
	IoU ↑	0.3706	0.4285	0.4348	0.4289	0.4202
Multi-view	LFD ↓	2378.48	2310.30	2413.18	2522.03	2639.69
	IoU ↑	0.6322	0.6595	0.6488	0.6317	0.6148
Voxels (32)	LFD ↓	1701.17	1683.95	1769.93	1900.48	2029.59
	IoU ↑	0.7636	0.7771	0.7659	0.7483	0.7323
Voxels (16)	LFD ↓	2453.69	2347.04	2426.40	2556.62	2724.72
	IoU ↑	0.6424	0.6726	0.6614	0.6452	0.6289
Points	LFD ↓	1429.37	1432.95	1521.55	1658.03	1830.78
	IoU ↑	0.8380	0.8379	0.8207	0.8002	0.7781

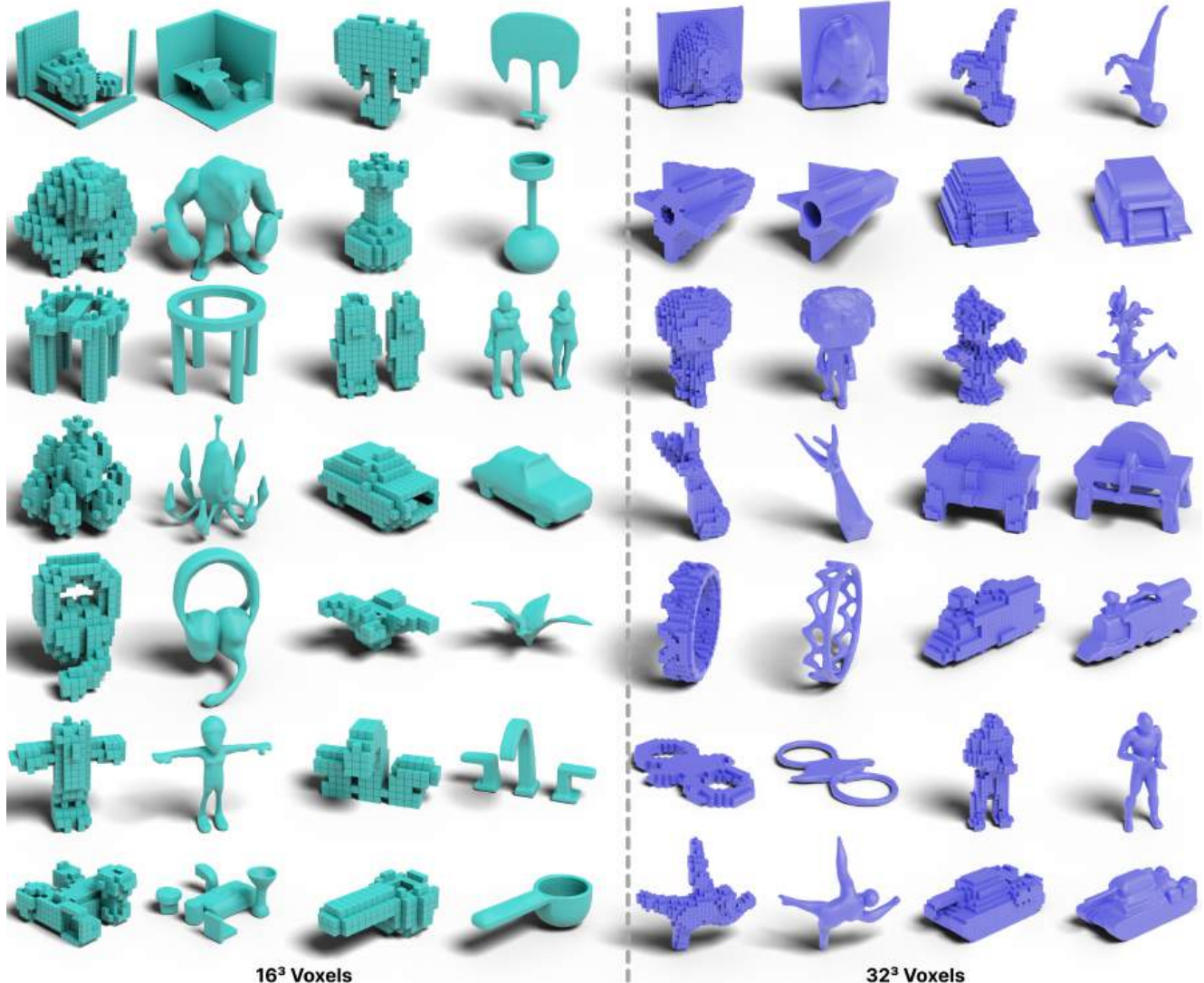


Figure 16. Our voxel-conditioned generative model excels in creating high-quality outputs from low-resolution inputs, imaginatively introducing various plausible geometric patterns. This is exemplified by the creation of holes in the crown, which are not available in the initial inputs.

is the scale parameter or guidance weight, denoted as w . This parameter plays a key role in managing the trade-off between the generation’s fidelity to the input conditions and the diversity of the generated output.

We experiment to explore the effect of the guidance weight parameter on the quality of samples generated by various conditional models. The guidance weight parameter was systematically adjusted in a linear progression from 1.0 to 5.0. It is important to note that, for efficient evaluation, an inference timestep of 100 was consistently employed across all experiments. The results of this study are presented in Table 6.

Empirically, we observe that a guidance weight of 2.0 is optimal for most conditional generation tasks. However,

when the model is conditioned on point cloud data, a lower guidance weight of 1.0 yields better results. This contrasts with the text-to-image scenarios, which typically require a larger value for the guidance weight. We suspect this difference is attributable to the nature of the input conditions we use, such as images and point clouds, which contain more information and thus make it more challenging to generate diverse samples compared to text-based inputs. Note that we adopt these identified optimal values as fixed hyperparameters for all subsequent inferences in the remainder of our experiments, as well as for the generation of qualitative results.

Inference Time Step Analysis. Furthermore, we also pro-

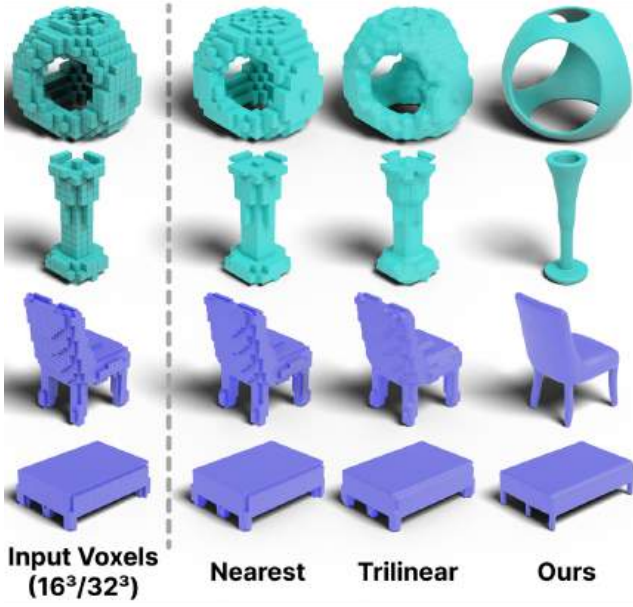


Figure 17. In comparison with meshes generated from interpolation using nearest neighbor upsampling and trilinear interpolation, our generation results display notably smoother surfaces.

Table 7. We quantitatively evaluate the performances of generation models with different inference time steps.

Model	Metrics	Inference Time step (t)			
		10	100	500	1000
Single-view	LFD ↓	4312.23	4071.33	4136.14	4113.10
	IoU ↑	0.4477	0.4285	0.4186	0.4144
Multi-view	LFD ↓	2217.25	2310.30	2369.15	2394.17
	IoU ↑	0.6707	0.6595	0.6514	0.6445
Voxels (32 ³)	LFD ↓	1580.98	1683.95	1744.48	1763.91
	IoU ↑	0.7943	0.7771	0.7700	0.7667
Voxels (16 ³)	LFD ↓	2266.41	2347.04	2375.89	2373.42
	IoU ↑	0.6870	0.6726	0.6620	0.6616
Point Cloud	LFD ↓	1368.90	1429.37	1457.89	1468.91
	IoU ↑	0.8535	0.8380	0.8283	0.8287
Unconditional	FID ↓	371.32	85.25	74.60	68.54

vide a detailed analysis of the inference timesteps for both our conditional and unconditional models. Specifically, we evaluate the generation models under the same settings as above but with varying timesteps, namely 10, 100, 500, and 1000.

Table 7 presents the quantitative results for our different generative models using various time steps during inference. Specifically, we empirically find that a small time step (10) suffices for conditions with minimal ambiguity, such as multi-view images, voxels, and point clouds. As ambiguity rises, the required number of time steps to achieve satisfactory sample quality also increases. For the unconditional

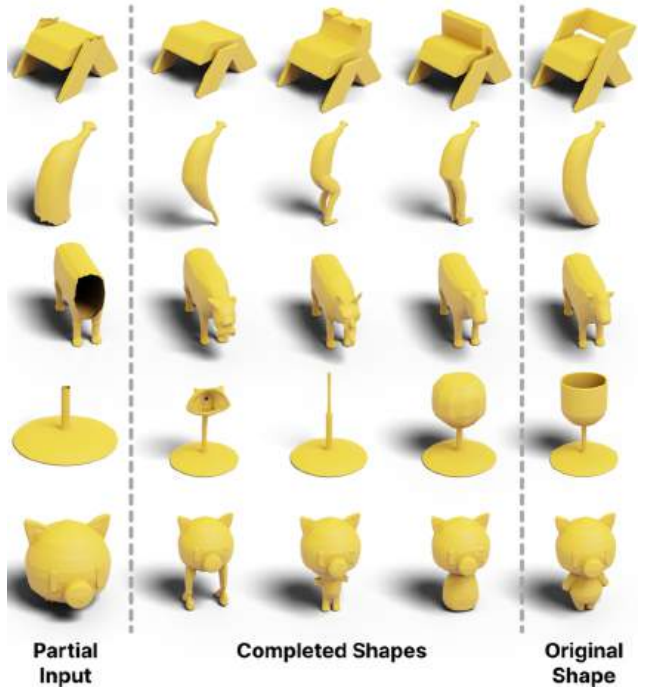


Figure 18. When provided with partial inputs (left), our unconditional generative model completes the missing regions in a coherent and semantically meaningful way. Additionally, it is capable of generating multiple variations of the completed shapes, many of which significantly differ from the original input (right), highlighting the diverse shape distribution learned.

Table 8. Employing Mean Squared Error (MSE) directly or separately for each subband resulted in worse performance compared to using just the coarse component [28], despite a higher theoretical representation capacity. In contrast, our subband adaptive training strategy led to significant improvements in both Light Field Distance (LFD) and Intersection Over Union (IoU) metrics.

Settings	Metrics	
	LFD ↓	IoU ↑
Coarse component only [28]	2855.41	0.5919
Ours (MSE)	3191.49	0.5474
Ours (subband-based MSE)	2824.28	0.5898
Ours	2611.60	0.6105

model, which has no condition, the optimal time step is the maximum one (1000). Similarly to the guidance weight, we consider the optimal time step as a hyper-parameter, which is utilized in all experiments.

Ablation of Adaptive Training Strategy. In this experiment, we use our multi-view model and initially compare our proposed representation with the generation results obtained using only the coarse component of our representation. This approach was adopted as the generation target

in [28]. This is shown in the first row of Table 8. Furthermore, to illustrate the effectiveness of our adopted subband adaptive training strategy, we also compare it with two other baseline training objectives. First, we apply a direct MSE (Mean Squared Error) loss uniformly across all our coefficients on our diffusible representation. This approach is equivalent to the objective of the classical diffusion model. Second, we contrast our strategy with a method that computes three separate MSE (Mean Squared Error) losses for C_0 , D_0 , and D_1 , and then sums them directly. It is important to note that separate models are necessary for each of these experiments, leading to increased computational resource costs. Additionally, convergence can take up to a few weeks, so we decided to implement an early stop at 750k iterations and compare the results at this stage.

From Table 8, it is observable that using solely the coarse component can still yield plausible results. This finding is in alignment with the discovery reported in [28]. However, naively applying MSE loss uniformly across all coefficients results in a performance drop, despite the upper bound of the representation capacity being significantly higher. We observe that the initial attempt to separate the loss computation improves the performance of the trained generator. However, the quality of the results remains similar to those generated using only the coarse component. By adopting our proposed subband adaptive training scheme, we achieve a significant improvement over the approach in [28].

9. Limitations and Future Works

Our approach exhibits the following limitations: (i) While our unconditional model is capable of generating a diverse variety of shapes from various sub-datasets, it can not ensure a balanced representation of objects across different categories during sampling. Hence, the learned 3D shape distribution is inherently imbalanced, evident in the disproportionate representation of CAD models. We can utilize a large zero-shot language model like ChatGPT for annotating object categories, enabling the application of diverse data augmentation methods to balance the training data according to these categories. (ii) Our generation network, trained on a heterogeneous mix of datasets, does not utilize the category label as an extra condition. Hence, our unconditional model may occasionally generate implausible shapes or introduce noise into the outputs. Identifying and mitigating these anomalies represents a compelling direction for future research. It is particularly intriguing to consider the development of data-driven metrics for assessing the visual plausibility of generated 3D shapes, especially in the context of the available large-scale 3D dataset. (iii) At present, our primary focus lies in direct generation of 3D geometry. An interesting avenue for future exploration involves generating textures together on the geometry, with the aim of achieving this without relying on

computationally-expensive optimizations.

10. Conclusion

In summary, this paper presents Make-A-Shape, a novel 3D generative framework trained on a vast dataset of over 10 millions publicly-available 3D shapes, capable of producing high-quality 3D shapes impressively within 2 seconds. Central to our approach is the introduction of a family of new techniques. This includes the subband coefficient filtering scheme to help construct a compact, expressive, and efficient wavelet-tree representation that effectively encodes a 256^3 SDF with minimal information loss. Then, we adeptly model the wavelet-tree representation by our diffusion-based generative model using our subband coefficient packing scheme, and further derive the subband adaptive training strategy to achieve model training that can effectively attends to both coarse and sparse detail coefficients. Besides, we also extend Make-A-Shape to take optional condition inputs of various modalities.

Our extensive experiments demonstrate the model’s superiority in synthesizing high-quality 3D shapes across various challenging conditions, including single/multi-view images, point clouds, and low-resolution voxels, all while requiring minimal resource demands during the training. Remarkably, our model not only outperforms existing baselines quantitatively but also demonstrates zero-shot applications such as partial shape completion. We believe our work will pave the way for future research in other 3D representations to enable large-scale 3D model training.

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. 4
- [2] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16123–16133, 2022. 4
- [3] 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. 10
- [4] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, pages 223–232. Wiley Online Library, 2003. 12
- [5] Qimin Chen, Zhiqin Chen, Hang Zhou, and Hao Zhang. Shaddr: Real-time example-based geometry and texture

- generation via 3d shape detailization and differentiable rendering. *arXiv preprint arXiv:2306.04889*, 2023. [5](#)
- [6] 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. [1](#), [4](#)
- [7] Zhiqin Chen, Vladimir G Kim, Matthew Fisher, Noam Aigerman, Hao Zhang, and Siddhartha Chaudhuri. Decorgan: 3d shape detailization by conditional refinement. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15740–15749, 2021. [5](#)
- [8] An-Chieh Cheng, Xueting Li, Sifei Liu, Min Sun, and Ming-Hsuan Yang. Autoregressive 3d shape generation via canonical mapping. In *European Conference on Computer Vision*, pages 89–104. Springer, 2022. [4](#)
- [9] Yen-Chi Cheng, Hsin-Ying Lee, Sergey Tulyakov, Alexander G Schwing, and Liang-Yan Gui. Sdfusion: Multimodal 3d shape completion, reconstruction, and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4456–4465, 2023. [4](#)
- [10] Gene Chou, Yuval Bahat, and Felix Heide. Diffusion-sdf: Conditional generative modeling of signed distance functions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2262–2272, 2023. [4](#)
- [11] Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F Yago Vicente, Thomas Dideriksen, Himanshu Arora, et al. Abo: Dataset and benchmarks for real-world 3d object understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21126–21136, 2022. [10](#)
- [12] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13142–13153, 2023. [5](#), [10](#), [12](#)
- [13] Congyue Deng, Chiyu Jiang, Charles R Qi, Xintan Yan, Yin Zhou, Leonidas Guibas, Dragomir Anguelov, et al. Nerdi: Single-view nerf synthesis with language-guided diffusion as general image priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20637–20647, 2023. [5](#)
- [14] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *NeurIPS*, 34:8780–8794, 2021. [10](#)
- [15] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2553–2560. IEEE, 2022. [3](#)
- [16] Huan Fu, Rongfei Jia, Lin Gao, Mingming Gong, Binqiang Zhao, Steve Maybank, and Dacheng Tao. 3d-future: 3d furniture shape with texture. *International Journal of Computer Vision*, 129:3313–3337, 2021. [10](#)
- [17] Chenjian Gao, Qian Yu, Lu Sheng, Yi-Zhe Song, and Dong Xu. Sketchsampler: Sketch-based 3d reconstruction via view-dependent depth sampling. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part I*, pages 464–479. Springer, 2022. [5](#)
- [18] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *Advances In Neural Information Processing Systems*, 35:31841–31854, 2022. [1](#), [4](#)
- [19] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. [4](#)
- [20] Benoit Guillard, Edoardo Remelli, Pierre Yvernay, and Pascal Fua. Sketch2mesh: Reconstructing and editing 3d shapes from sketches. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13023–13032, 2021. [5](#)
- [21] Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. Meshcnn: a network with an edge. *ACM Transactions on Graphics (ToG)*, 38(4):1–12, 2019. [4](#)
- [22] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. [12](#)
- [23] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. [9](#), [16](#)
- [24] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. [3](#), [7](#)
- [25] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. *arXiv preprint arXiv:2311.04400*, 2023. [3](#), [4](#)
- [26] Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. *arXiv preprint arXiv:2301.11093*, 2023. [3](#), [8](#), [10](#)
- [27] Zhongzhan Huang, Pan Zhou, Shuicheng Yan, and Liang Lin. Scalelong: Towards more stable training of diffusion model via scaling network long skip connection. *arXiv preprint arXiv:2310.13545*, 2023. [10](#)
- [28] Ka-Hei Hui, Ruihui Li, Jingyu Hu, and Chi-Wing Fu. Neural wavelet-domain diffusion for 3D shape generation. In *ACM SIGGRAPH Asia*, pages 1–9, 2022. [1](#), [3](#), [4](#), [5](#), [6](#), [18](#), [19](#)
- [29] 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. [4](#)
- [30] Ajay Jain, Ben Mildenhall, Jonathan T Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 867–876, 2022. [5](#)

- [31] Pradeep Kumar Jayaraman, Aditya Sanghi, Joseph G Lambourne, Karl DD Willis, Thomas Davies, Hooman Shayani, and Nigel Morris. Uv-net: Learning from boundary representations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11703–11712, 2021. 4
- [32] Pradeep Kumar Jayaraman, Joseph G Lambourne, Nishkrit Desai, Karl DD Willis, Aditya Sanghi, and Nigel JW Morris. Solidgen: An autoregressive model for direct b-rep synthesis. *arXiv preprint arXiv:2203.13944*, 2022. 1, 4
- [33] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3D implicit functions. *arXiv preprint arXiv:2305.02463*, 2023. 3, 4, 5, 10, 11, 12
- [34] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 10
- [35] 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. 4
- [36] Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. Abc: A big cad model dataset for geometric deep learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9601–9611, 2019. 10
- [37] Di Kong, Qiang Wang, and Yonggang Qi. A diffusion-refinement model for sketch-to-point modeling. In *Proceedings of the Asian Conference on Computer Vision*, pages 1522–1538, 2022. 5
- [38] Joseph G Lambourne, Karl DD Willis, Pradeep Kumar Jayaraman, Aditya Sanghi, Peter Meltzer, and Hooman Shayani. Brepnet: A topological message passing system for solid models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12773–12782, 2021. 4
- [39] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pages 3744–3753. PMLR, 2019. 10, 14
- [40] Jiahao Li, Hao Tan, Kai Zhang, Zexiang Xu, Fujun Luan, Yinghao Xu, Yicong Hong, Kalyan Sunkavalli, Greg Shakhnarovich, and Sai Bi. Instant3d: Fast text-to-3d with sparse-view generation and large reconstruction model. *arXiv preprint arXiv:2311.06214*, 2023. 3, 4
- [41] Muheng Li, Yueqi Duan, Jie Zhou, and Jiwen Lu. Diffusion-sdf: Text-to-shape via voxelized diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12642–12651, 2023. 4
- [42] Yang Li, Hikari Takehara, Takafumi Taketomi, Bo Zheng, and Matthias Nießner. 4dcomplete: Non-rigid motion estimation beyond the observable surface. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12706–12716, 2021. 10
- [43] Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Zexiang Xu, Hao Su, et al. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. *arXiv preprint arXiv:2306.16928*, 2023. 3, 4, 10, 11, 12
- [44] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9298–9309, 2023. 5, 12
- [45] Xinhai Liu, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker. Fine-grained 3d shape classification with hierarchical part-view attention. *IEEE Transactions on Image Processing*, 30:1744–1758, 2021. 10
- [46] Zhengzhe Liu, Peng Dai, Ruihui Li, Xiaojuan Qi, and Chi-Wing Fu. Iss: Image as setting stone for text-guided 3d shape generation. *arXiv preprint arXiv:2209.04145*, 2022. 5
- [47] Zhengzhe Liu, Jingyu Hu, Ka-Hei Hui, Xiaojuan Qi, Daniel Cohen-Or, and Chi-Wing Fu. Exim: A hybrid explicit-implicit representation for text-guided 3d shape generation. *ACM Transactions on Graphics (TOG)*, 42(6):1–12, 2023. 4
- [48] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, 2015. 10
- [49] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Seminal graphics: pioneering efforts that shaped the field*, pages 347–353. 1998. 15
- [50] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471, 2022. 15
- [51] Zhaoliang Lun, Matheus Gadelha, Evangelos Kalogerakis, Subhransu Maji, and Rui Wang. 3d shape reconstruction from sketches via multi-view convolutional networks. In *2017 International Conference on 3D Vision (3DV)*, pages 67–77. IEEE, 2017. 5
- [52] 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. 4, 5
- [53] Jonathan Masci, Davide Boscaini, Michael Bronstein, and Pierre Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 37–45, 2015. 4
- [54] 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. 4
- [55] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. Realfusion: 360deg reconstruction of any object from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8446–8455, 2023. 5

- [56] 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. 1, 4
- [57] Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2mesh: Text-driven neural stylization for meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13492–13502, 2022. 5
- [58] Aryan Mikaeili, Or Perel, Mehdi Safaei, Daniel Cohen-Or, and Ali Mahdavi-Amiri. Sked: Sketch-guided text-based 3d editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14607–14619, 2023. 5
- [59] 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. 4
- [60] Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy Mitra, and Leonidas J Guibas. Structurenets: Hierarchical graph networks for 3d shape generation. *arXiv preprint arXiv:1908.00575*, 2019. 4
- [61] 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. 4
- [62] Alex Nichol, Heewoo Jun, Pratul Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022. 3, 4, 5, 11, 12
- [63] 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. 4
- [64] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 523–540. Springer, 2020. 4
- [65] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 5
- [66] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656, 2016. 4
- [67] 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. 4, 9, 14
- [68] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 4
- [69] Guocheng Qian, Jinjie Mai, Abdullah Hamdi, Jian Ren, Aliaksandr Siarohin, Bing Li, Hsin-Ying Lee, Ivan Skokhodov, Peter Wonka, Sergey Tulyakov, et al. Magic123: One image to high-quality 3d object generation using both 2d and 3d diffusion priors. *arXiv preprint arXiv:2306.17843*, 2023. 5
- [70] 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. 9
- [71] Alexander Raistrick, Lahav Lipson, Zeyu Ma, Lingjie Mei, Mingzhe Wang, Yiming Zuo, Karhan Kayan, Hongyu Wen, Beining Han, Yihan Wang, et al. Infinite photorealistic worlds using procedural generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12630–12641, 2023. 10
- [72] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021. 1
- [73] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J. Black. Generating 3D faces using convolutional mesh autoencoders. In *European Conference on Computer Vision (ECCV)*, pages 725–741, 2018. 10
- [74] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 1, 5, 10
- [75] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022. 1, 5
- [76] Aditya Sanghi, Hang Chu, Joseph G. Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshahi. CLIP-Forge: Towards zero-shot text-to-shape generation. In *CVPR*, pages 18603–18613, 2022. 4, 5
- [77] Aditya Sanghi, Rao Fu, Vivian Liu, Karl DD Willis, Hooman Shayani, Amir H Khasahmadi, Srinath Sridhar, and Daniel Ritchie. Clip-sculptor: Zero-shot generation of high-fidelity and diverse shapes from natural language. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18339–18348, 2023. 4, 5
- [78] Aditya Sanghi, Pradeep Kumar Jayaraman, Arianna Rampini, Joseph Lambourne, Hooman Shayani, Evan Atherton, and Saeid Asgari Taghanaki. Sketch-a-shape:

- Zero-shot sketch-to-3d shape generation. *arXiv preprint arXiv:2307.03869*, 2023. 5
- [79] Katja Schwarz, Axel Sauer, Michael Niemeyer, Yiyi Liao, and Andreas Geiger. Voxgraf: Fast 3d-aware image synthesis with sparse voxel grids. *Advances in Neural Information Processing Systems*, 35:33999–34011, 2022. 4
- [80] Pratheba Selvaraju, Mohamed Nabail, Marios Loizou, Maria Maslioukova, Melinos Averkiou, Andreas Andreou, Siddhartha Chaudhuri, and Evangelos Kalogerakis. Buildingnet: Learning to label 3d buildings. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10397–10407, 2021. 10
- [81] Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation. *arXiv preprint arXiv:2308.16512*, 2023. 5
- [82] J Ryan Shue, Eric Ryan Chan, Ryan Po, Zachary Ankner, Jiajun Wu, and Gordon Wetzstein. 3D neural field generation using triplane diffusion. In *CVPR*, pages 20875–20886, 2023. 1, 4
- [83] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015. 3
- [84] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019. 3
- [85] Stefan Stojanov, Anh Thai, and James M Rehg. Using shape to categorize: Low-shot learning with an explicit shape bias. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1798–1808, 2021. 10
- [86] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015. 4
- [87] 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. 4
- [88] Nitika Verma, Edmond Boyer, and Jakob Verbeek. Feastnet: Feature-steered graph convolutions for 3d shape analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2598–2606, 2018. 4
- [89] Kashi Venkatesh Vishwanath, Diwaker Gupta, Amin Vahdat, and Ken Yocum. Modelnet: Towards a datacenter emulation environment. In *2009 IEEE Ninth International Conference on Peer-to-Peer Computing*, pages 81–82. IEEE, 2009. 10
- [90] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12, 2019. 4
- [91] Karl DD Willis, Yewen Pu, Jieliang Luo, Hang Chu, Tao Du, Joseph G Lambourne, Armando Solar-Lezama, and Wojciech Matusik. Fusion 360 gallery: A dataset and environment for programmatic cad construction from human design sequences. *ACM Transactions on Graphics (TOG)*, 40(4):1–24, 2021. 10
- [92] 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. 4
- [93] Rundi Wu, Chang Xiao, and Changxi Zheng. Deepcad: A deep generative network for computer-aided design models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6772–6782, 2021. 4
- [94] Yi Wu, Yuxin Wu, Georgia Gkioxari, and Yuandong Tian. Building generalizable agents with a realistic and rich 3d environment. *arXiv preprint arXiv:1801.02209*, 2018. 10
- [95] 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. 4
- [96] Dejia Xu, Yifan Jiang, Peihao Wang, Zhiwen Fan, Yi Wang, and Zhangyang Wang. Neurallift-360: Lifting an in-the-wild 2d photo to a 3d object with 360 views. *arXiv e-prints*, pages arXiv–2211, 2022. 5
- [97] Jiale Xu, Xintao Wang, Weihao Cheng, Yan-Pei Cao, Ying Shan, Xiaohu Qie, and Shenghua Gao. Dream3d: Zero-shot text-to-3d synthesis using 3d shape prior and text-to-image diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20908–20918, 2023. 5
- [98] Yinghao Xu, Hao Tan, Fujun Luan, Sai Bi, Peng Wang, Jiahao Li, Zifan Shi, Kalyan Sunkavalli, Gordon Wetzstein, Zexiang Xu, et al. Dmv3d: Denoising multi-view diffusion using 3d large reconstruction model. *arXiv preprint arXiv:2311.09217*, 2023. 3, 4
- [99] 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. 4
- [100] 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. 1, 4
- [101] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2(3):5, 2022. 1
- [102] 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
- [103] Biao Zhang, Matthias Nießner, and Peter Wonka. 3dirl: Irregular latent grids for 3d generative modeling. *Advances in*

Neural Information Processing Systems, 35:21871–21885, 2022. 4, 5

- [104] Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *arXiv preprint arXiv:2301.11445*, 2023. 4, 12
- [105] Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3DShape2VecSet: A 3D shape representation for neural fields and generative diffusion models. 42(4), 2023. 3, 5
- [106] Xinyang Zheng, Yang Liu, Pengshuai Wang, and Xin Tong. Sdf-stylegan: Implicit sdf-based stylegan for 3d shape generation. In *Computer Graphics Forum*, pages 52–63. Wiley Online Library, 2022. 4
- [107] Xin-Yang Zheng, Hao Pan, Peng-Shuai Wang, Xin Tong, Yang Liu, and Heung-Yeung Shum. Locally attentional sdf diffusion for controllable 3d shape generation. *arXiv preprint arXiv:2305.04461*, 2023. 4, 5
- [108] 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. 4, 5
- [109] Qingnan Zhou and Alec Jacobson. Thingi10k: A dataset of 10,000 3d-printing models. *arXiv preprint arXiv:1605.04797*, 2016. 10
- [110] Silvia Zuffi, Angjoo Kanazawa, David Jacobs, and Michael J. Black. 3D menagerie: Modeling the 3D shape and pose of animals. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 10