# 3DTopia-XL: Scaling High-quality 3D Asset Generation via Primitive Diffusion
## –*Supplementary Material*–

Zhaoxi Chen[1] Jiaxiang Tang[2] Yuhao Dong[1,3] Ziang Cao[1]
Fangzhou Hong[1] Yushi Lan[1] Tengfei Wang[3] Haozhe Xie[1]
Tong Wu[3,4] Shunsuke Saito Liang Pan[3] Dahua Lin[3,4] ✉ Ziwei Liu[1] ✉
[1] S-Lab, Nanyang Technological University [2] Peking University
[3] Shanghai AI Laboratory [4] The Chinese University of Hong Kong

https://3dtopia.github.io/3DTopia-XL/

## Contents



# A. Appendix

This supplementary material is organized as follows:

- Sec. A.1 provides further discussions, including the main difference between PrimX and existing 3D representations (Sec. A.1.1) and limitations (Sec. A.1.2).
- Sec. A.2 introduces further experiments and evaluations, including quantitative results on Objaverse [5] and GSO [6] datasets (Sec. A.2.1), user study (Sec. A.2.3), model scaling (Sec. A.2.4), sampling diversity (Sec. A.2.5), additional ablation studies on PrimX initialization (Sec. A.2.7), VAE designs (Sec. A.2.8), PBR extraction (Sec. A.2.9), differentiability (Sec. A.2.10) and more qualitative results (Sec. A.2.11).
- Sec. A.3 documents the implementations details of 3DTopia-XL, including dataset and PrimX hyperparameters (Sec. A.3.1), conditioner and captions (Sec. A.3.2), model details and hyperparameters (Sec. A.3.3), and algorithms of reversible conversion between PrimX and mesh (Sec. A.3.4).
- Besides, we also attach a **demo video** to demonstrate the key idea and qualitative results.

## A.1. Discussion

### A.1.1. Difference with Related Work

The core of our work is the proposed novel 3D representation, PrimX, that can model high-quality 3D shape, texture, and material in a unified and tensorial representation. It is worth highlighting the advantages of PrimX compared with other 3D representations in the **generative context**.

**PrimX v.s. Implicit Vector Set.** Previous works [34, 35] introduce the implicit vector set to encode a 3D shape globally. PrimX differentiates itself from the implicit vector set in three aspects:

- PrimX encodes not only the shape but also texture and material in a unified way, which removes the necessity for a two-stage framework [35] that generates shape and texture separately. Although the implicit vector set has the potential to model more information other than the occu-

Table 1. **Quantitative evaluations of Image-to-3D on the Objaverse dataset [5].** We evaluate the fidelity, diversity and distributional similarity of 3DTopia-XL for single image to 3D generation compared with existing baselines. The evaluation is performed on a subset of Objaverse consisting of 600 random samples. $\text{KID}_{\text{mat}}$ denotes the KID measured in the material space by rendering materials as colored 2D images. FPD and KPD denote the FID and KID metrics measured on 3D points in the space of PointNet++ [20]. COV and MMD denote Coverage Score and Minimum Matching Distance measured in the space of Chamfer Distance.

| Method | Paradigm | KID ($\times 10^{-2}$) ↓ | $\text{KID}_{\text{mat}}$ ($\times 10^{-2}$) ↓ | FPD ↓ | KPD ($\times 10^{-2}$) ↓ | COV ↑ | MMD ($\times 10^{-3}$) ↓ |
|---|---|---|---|---|---|---|---|
| LGM [26] | Sparse-view Reconstruction | 0.55 | - | 39.86 | 18.54 | 35.83 | 19.88 |
| CRM [29] | Sparse-view Reconstruction | 1.93 | - | 37.09 | 14.37 | 31.83 | 25.12 |
| ShapE [10] | Native 3D Diffusion | 11.95 | - | 20.27 | 7.55 | 59.67 | 14.98 |
| LN3Diff [14] | Native 3D Diffusion | 0.89 | - | 29.36 | 11.27 | 33.50 | 22.64 |
| **Ours** | Native 3D Diffusion | 1.02 | 0.69 | 15.74 | 3.59 | 50.31 | 14.63 |

Table 2. **Quantitative evaluations of Image-to-3D on the GSO dataset [6].** We evaluate the fidelity, diversity and distributional similarity of 3DTopia-XL for single image to 3D generation compared with existing baselines. The evaluation is performed on a subset of GSO dataset consisting of 300 random samples. FPD and KPD denote the FID and KID metrics measured on 3D points in the space of PointNet++ [20]. COV and MMD denote Coverage Score and Minimum Matching Distance measured in the space of Chamfer Distance.

| Method | Paradigm | KID ($\times 10^{-2}$) ↓ | FPD ↓ | KPD ($\times 10^{-2}$) ↓ | COV ↑ | MMD ($\times 10^{-3}$) ↓ |
|---|---|---|---|---|---|---|
| LGM [26] | Sparse-view Reconstruction | 0.94 | 34.44 | 11.12 | 33.11 | 23.32 |
| CRM [29] | Sparse-view Reconstruction | 1.67 | 26.61 | 4.82 | 38.57 | 17.37 |
| InstantMesh [32] | Sparse-view Reconstruction | 0.91 | 21.54 | 4.15 | 37.01 | 16.31 |
| ShapE [10] | Native 3D Diffusion | 13.75 | 22.80 | 4.03 | 40.43 | 18.58 |
| **Ours** | Native 3D Diffusion | 1.38 | 16.16 | 3.86 | 55.58 | 14.35 |

pancy field, there is no existing work to demonstrate and justify this design. We suspect that texture and material information are surface-aligned which is far more expensive and difficult to model for the implicit vector set that uses dense modeling of the entire 3D space by using 3D points as queries.
- PrimX is differentiable renderable while implicit vector set can be only exported to meshes.
- PrimX is explicit and explainable for each token feature which facilitates 1) data augmentation by applying color transformation similar to [11]; and 2) downstream tasks like inpainting by explicitly masking certain tokens.

**PrimX v.s. PrimDiffusion [3].** Chen et al. [3] proposes using volumetric primitives for 3D human generation, learning primitive-based representation from multi-view posed images. PrimX has unique differences:
- PrimX requires no 3D template for generative modeling by directly denoising the position of primitives. In contrast, PrimDiffusion requires a template mesh as the anchor for all primitives which works for 3D human generation where the target subject has a shared 3D canonical space. However, this is not the case for general objects as there is no mesh template.
- PrimX simplifies the parameter space of volumetric primitives by using only position, a single scale, and voxelized payload, while the prior work models per-axis rotation and scale factors additionally. This simplification significantly saves the computational cost while achieving a comparable quality in our preliminary study.
- PrimX models the target's geometry as SDF field and is capable of learning from both 3D data and 2D data. The work above models the target's geometry as volumetric opacity field and can only learn from 2D images.

**PrimX v.s. M-SDF [33].** M-SDF introduces a shape-only representation to encode SDF of 3D mesh into mosaic voxels. PrimX has two distinct differences compared to it:
- M-SDF only represents 3D shape, while our method finds a unified way to encode shape, texture, and material with high quality. It is non-trivial to represent shape, texture, and material within our tensorial and sparse representation. The shape is typically a 3D volumetric function while textural information is only surface-aligned. It is important to note that 1) proper instantiation of texture sampling function (Sec. A.3.1) and 2) carefully designed initialization strategy (Alg. 1) for PrimX are critical for representing shape, texture and material in high quality.
- M-SDF is specialized to 3D domain while our representation can be differentiably rendered into 2D images.

**PrimX v.s. 3DGS [12].** As a trending representation for 3D reconstruction, 3DGS is known for its efficiency as a primitive-based volumetric representation. However, the

number of Gaussians required to represent a high-quality 3D object is considerably high (hundreds of thousands) compared with PrimX (N=2048). This long context property will lead to training difficulty and inefficient attention computation in the generative context where the set of Gaussians is operated by DiT [19]. Instead, PrimX can be treated as an "interpolation" between fully point-based representation (3DGS) and fully voxel-based representation (dense voxel) that groups primitives into explicitly structured local voxels. This hybrid operation significantly reduces the number of primitives, leading to a shorter context that boosts the training of the Transformer.

#### A.1.2. Limitations and Future Work

It is important to note that 3DTopia-XL has been trained on a considerably large-scale dataset. However, there is still room for improvement in terms of quality. Different from existing high-quality 3D diffusion models [33, 35] which operate on 3D representations that are not differentiably renderable, 3DTopia-XL maintains the ability to directly learn from 2D image collections thanks to PrimX's capability of differentiable rendering (Eq. 7 in the main paper). This opens up new opportunities to learn 3D generative models from a mixture of 3D and 2D data, which can be a solution to the lack of high-quality 3D data. Moreover, as an explicit representation, PrimX is interpretable and easy to drive. By manipulating primitives or groups of primitives, it is also fruitful to explore dynamic object generation and generative editing.

### A.2. Additional Experiments

#### A.2.1. Quantitative Comparisons on Objaverse and GSO

We conduct extensive quantitative evaluations for image-to-3D on Objaverse [5] and GSO [6] datasets against existing methods including: 1) sparse-view reconstruction model: LGM [26], CRM [29], InstantMesh [32] and 2) native 3D diffusion models: ShapE [10] and LN3Diff [14].

**Evaluation Metrics.** For photometric quality, we benchmark KID [1] over the 2D renderings under random environmental lighting against ground truth. $\text{KID}_{\text{mat}}$ is also calculated to measure the quality of the generated PBR materials. We render the metallic and roughness into colored 2D images according to gltf 2.0 specifics[1], and measure KID against ground truth. For geometric quality, we measure Point Cloud FID [7] (FPD) and Point Cloud KID [1] using the pretrained PointNet++ [20] provided by following previous work [10, 33]. Moreover, we also evaluate Coverage Score (COV) and Minimum Matching Distance (MMD) in the space of Chamfer Distance (CD) following previous work [33, 34]. We perform the farthest point sampling over the output mesh for each method to obtain 4096 points for evaluation. We randomly sample 300 objects on the GSO dataset and 600 objects on the Objaverse dataset for evaluation, respectively.

**Results.** As shown in Figure 1 and Figure 2, our method achieves the best 3D geometric quality, indicating the superiority of the proposed representation and generative modeling. Note that the methods based on sparse-view reconstruction rely on pretrained 2D multiview diffusion models [24, 28] to reconstruct 3D objects from sparse input views. Therefore, their models have more input visual information and thus achieve slightly better visual quality. However, this cascaded pipeline prone to yielding 3D distortions due to 3D inconsistency of 2D diffusion models, indicated by the worse 3D metrics of them. Most importantly, 3DTopia-XL is the only method capable of producing PBR materials from images or texts among all methods.

#### A.2.2. Additional Comparisons

We show image-to-3D comparisons with Unique3D [30] in Fig. 3(c). As a per-sample optimization-based method using multiview images, Unique3D excels in texture quality but suffers from geometry artifacts (weird geometry from novel views) due to the inconsistency of 2D diffusion models. We believe that training a feedforward reconstruction method with PrimX would be a good geometry initialization and mesh constraint for Unique3D, which shortens its optimization time.

Moreover, we demonstrate our image-to-3D generation using a casual phone capture in Fig. 3(b), indicating the generalizability of 3DTopia-XL to the real-world domain.

#### A.2.3. User Study

We conduct an extensive user study to evaluate image-to-3D performance quantitatively. We opt for an output evaluation [2] for user study, where each volunteer is shown with a pair of results comparing a random method against ours, and asked to choose the better one in four aspects: **1)** Overall Quality, **2)** Image Alignment, **3)** Surface Smoothness, and **4)** Physical Correctness. One of the samples presented to the attendees is shown in Figure 5. A total number of 48 paired samples are provided to 27 volunteers for the flip test. We summarize the average preference percentage across all four dimensions in Figure 4. 3DTopia-XL is the best one among all methods. Although the image alignment of our method is only a slight improvement against reconstruction-based methods like CRM, the superior quality of geometry and the ability to model physically based materials are the keys to producing the best overall quality in the final rendering.

#### A.2.4. Scaling

We further investigate the scaling law of 3DTopia-XL against model sizes and iterations. For metrics, we use Fréchet Inception Distance (FID) computed over 5k ran-

[1] https://registry.khronos.org/glTF/specs/2.0/glTF-2.0.html#metallic-roughness-material


Figure 1. **3DTopia-XL can generate 3D assets directly from single-view images without relying on 2D image-to-multiview diffusion models.** We visualize the input single-view image and corresponding HDRIs for environmental lighting on the left. Please note the high-quality results and spatially varied materials generated by our method. All scenes are rendered using Blender [4].

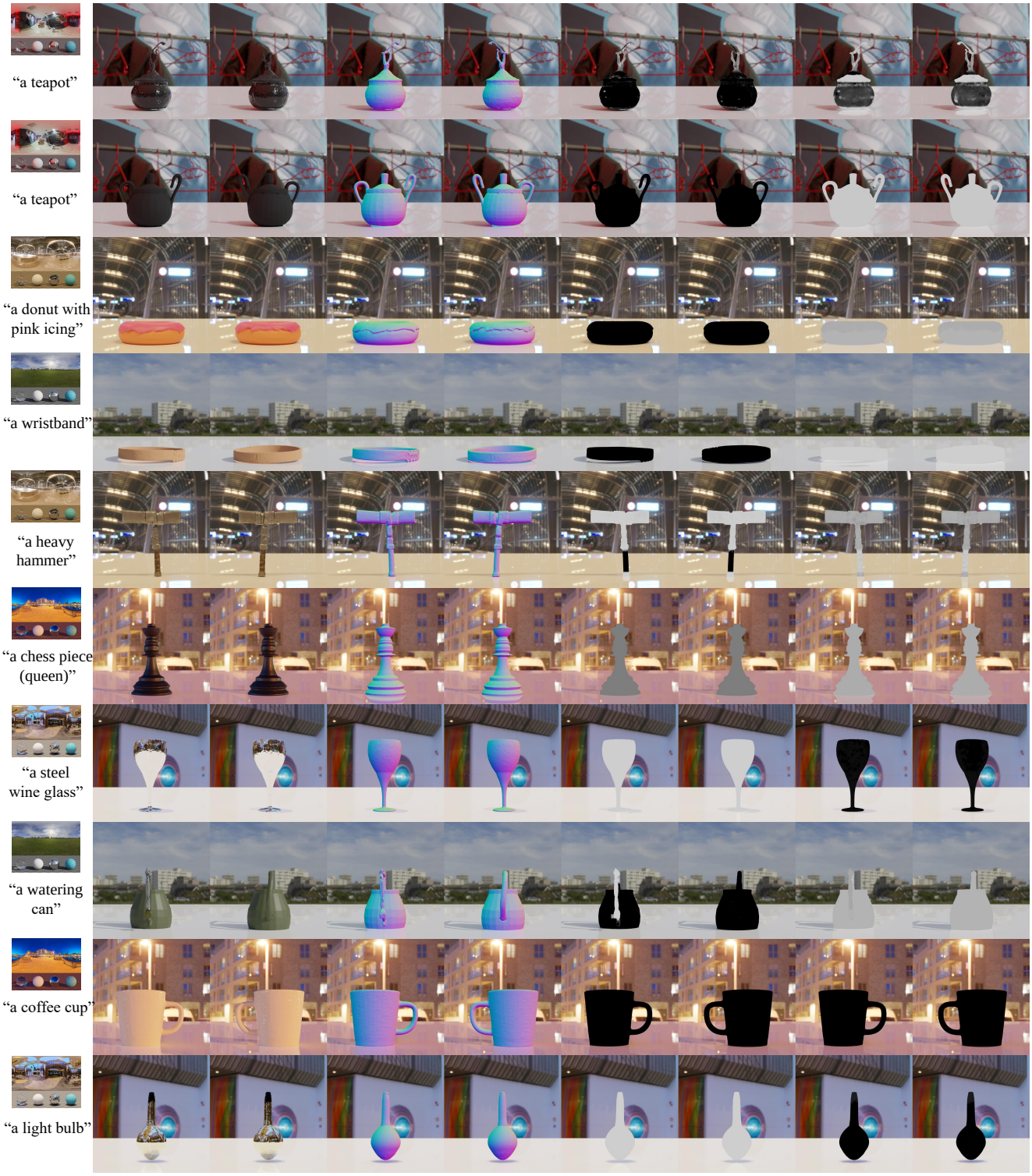


Figure 2. **3DTopia-XL can generate 3D assets directly from texts without relying on 2D text-to-image diffusion models, which is uniquely different from sparse-view reconstruction models [9].** We visualize the input text prompts and corresponding HDRIs for environmental lighting on the left. Please note the sampling diversity and spatially varied materials generated by our method. All scenes are rendered using Blender [4].

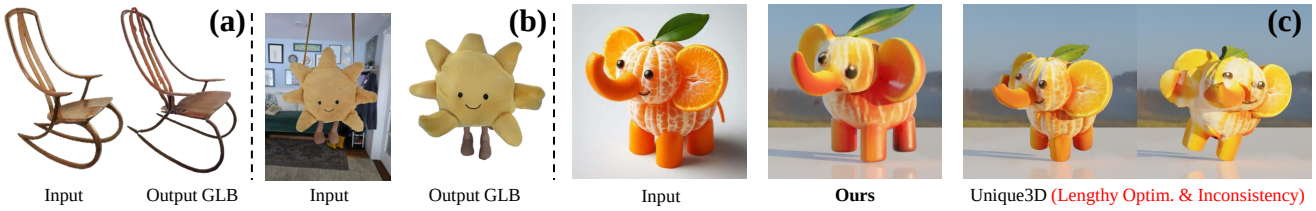


Figure 3. **(a)** Thin structures. **(b)** Real-world inputs. **(c)** Comparison with Unique3D, which shows geometry inconsistency.

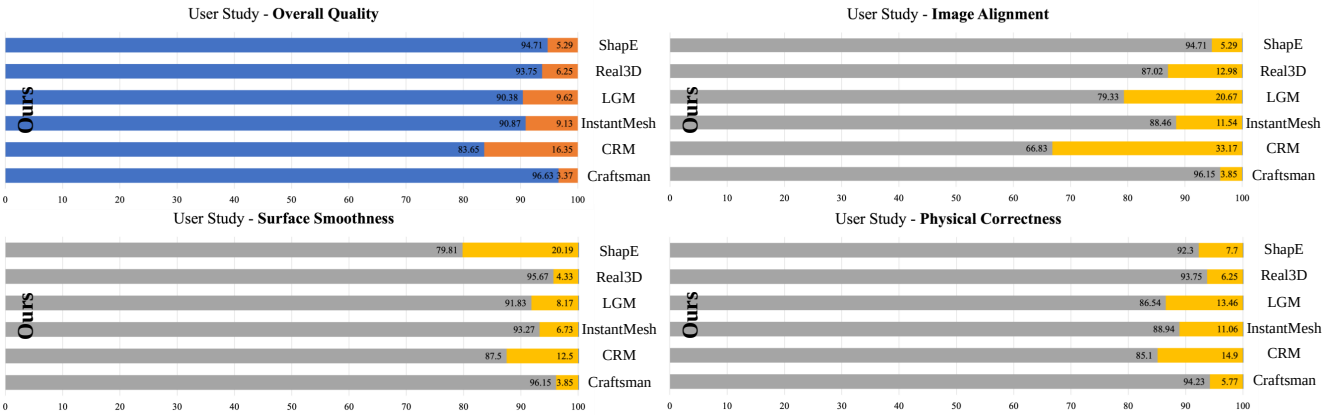


Figure 4. **User study.** We quantitatively evaluate comparison methods by conducting preference tests against our method on four dimensions. The results show that 3DTopia-XL has the highest preference rate compared with other methods.

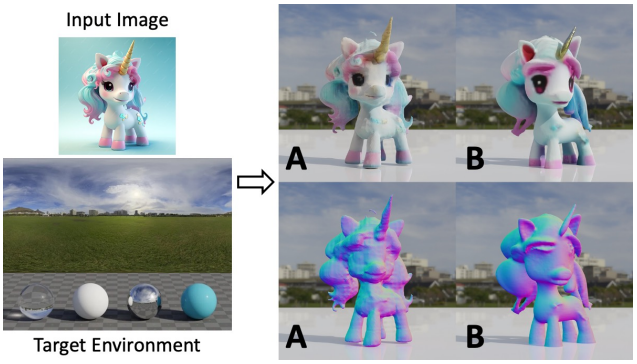


Figure 5. **User study sample.** For each sample in the user study, we present to the attendee with the input image (upper left) and target environment illuminations (bottom left) for rendering the mesh. Each volunteer is asked to choose the better one from A/B across four dimensions: 1) Overall quality, 2) Image alignment, 3) Surface smoothness, and 4) Physical correctness of renderings. The order and notation of methods are randomized and anonymized.

dom samples without CFG guidance. Specifically, we consider Latent-FID which is computed in the latent space of our VAE and Rendering-FID which is computed on the DINO [18] embeddings extracted from images rendered with Eq. 7 in the main paper. Figure 6 shows how Latent-FID and Rendering-FID change as the model size increases. We observe consistent improvements as the model becomes deeper and wider. Table 3 also demonstrates that longer sequence (smaller patches) leads to better performance, which may come from the findings in the vanilla DiT that increasing GFlops leads to better performance.

Table 3. **Longer sequence leads to better convergence.** Given a fixed PrimX parameter budget of 1.05M, we compare the models trained with $\{N = 256, a = 16\}$ and $\{N = 2048, a = 8\}$.

| Setting | Rendering-FID ↓ | Latent-FID ↓ |
|---|---|---|
| $N = 256$ | 76.31 | 104.8 |
| $N = 2048$ | 16.16 | 24.43 |

### A.2.5. Sampling Diversity

At last, we demonstrate the impressive sampling diversity of 3DTopia-XL as a generative model, as shown in Figure 7. Given the same input image and varying random seeds, our model can generate diverse high-quality 3D assets with different geometry and spatially varied PBR materials.

### A.2.6. Generation of Inner Geometric Structures

We present results with plausible and diverse inner structures in Fig. 8. Thanks to the native 3D representation PrimX, 3DTopia-XL can generate well-defined and diverse

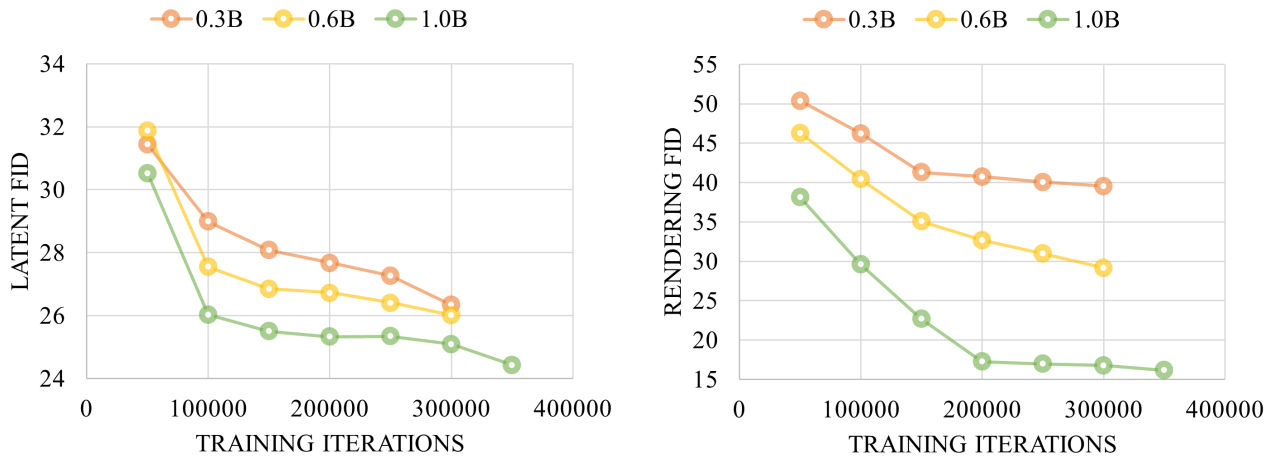


Figure 6. **Scaling up 3DTopia-XL improves FID.** As the computation and model size scale up, the model performance improves consistently. For metrics, we consider Latent-FID which is computed in the latent space of our VAE and Rendering-FID which is computed on the DINO [18] embeddings extracted from images.

inner structures from images / texts, which facilitates downstream tasks such as physical simulations and compositional object generation. Additional results on generating thin structure is shown in Fig. 3(a).

### A.2.7. Ablation study on PrimX Initialization

In this section, we conduct ablation studies on the impact of different initialization strategies for mesh to PrimX conversion (Algorithm 1). We compare three alternatives here:

- **Uniform + Farthest (Ours)**: 1) we first perform uniform sampling to get $\hat{N}$ candidate points; and 2) we run farthest point sampling on the candidate point set to get $N$ primitives and initialize their scales to ensure coverage.
- **Farthest**: directly perform farthest point sampling to get $N$ primitives with a unique global scale factor as in M-SDF [33].
- **Coverage**: 1) we first perform farthest point sampling to get $\frac{3}{4}N$ primitives; 2) a uniformly sampled point set is used to test the coverage by existing primitives, and points not covered are held out; and 3) we perform the second farthest point sampling on the held-out set to get the rest $\frac{1}{4}N$ primitive.

As shown in Figure 9, the "Farthest" solution is sensitive to the topology, which may lead to the insufficient number of primitive allocated to the flattened surface with a few mesh faces, causing the gap in the drill. Our final solution achieves comparable quality with the complicated "Coverage" solution and is capable of modeling fine-grained geometric details and consistent texture and material with ground truth. However, due to unnecessary computation overhead introduced by the latter solution, we choose the "Uniform + Farthest" initialization strategy as the final solution which is simple but effective. Quantitative results in Table 4 also confirm the above observation.

### A.2.8. Ablation study on VAE Designs

Given our goal to do spatial compression of VAE in Primitive Patch Compression, we additionally compare two alternatives suitable for spatial compression:

- **PCA:** Principal Component Analysis uses a certain number of principal components and projects PrimX onto a low-dimensional space. Take the projection matrix consisting of principal components as $\boldsymbol{P}$, the encoding process denotes as $\boldsymbol{XP}$ while the decoding process denotes as $\boldsymbol{XPP}^T$. This compression mechanism relies on a set of known principal components at inference time.
- **DCT:** Discrete Cosine Transformation similar to the JPEG [27] compression algorithm. As our PrimX is naturally divided by 8 for each primitive, we can traverse through each voxel's value in zig-zag order and perform DCT for spatial compression.

As shown in Figure 10, PCA fails to reproduce smooth geometry at all compression rates due to the loss of spatial structure during the encoding process. DCT, as a widely used spatial compression algorithm for 2D images, can also achieve reasonably good compression quality at a relatively lower compression rate (3072 to 384). However, it fails at a higher compression rate and cannot achieve comparable quality as patch-wise VAE.

### A.2.9. Ablation study on PrimX2Mesh Algorithm

As we mentioned in the main paper existing work on 3D generation did not carefully deal with mesh extraction when converting the underlying 3D representations to GLB formatted meshes. In this paper, we carefully design the PrimX2Mesh algorithm as outlined in Alg. 2. Furthermore,

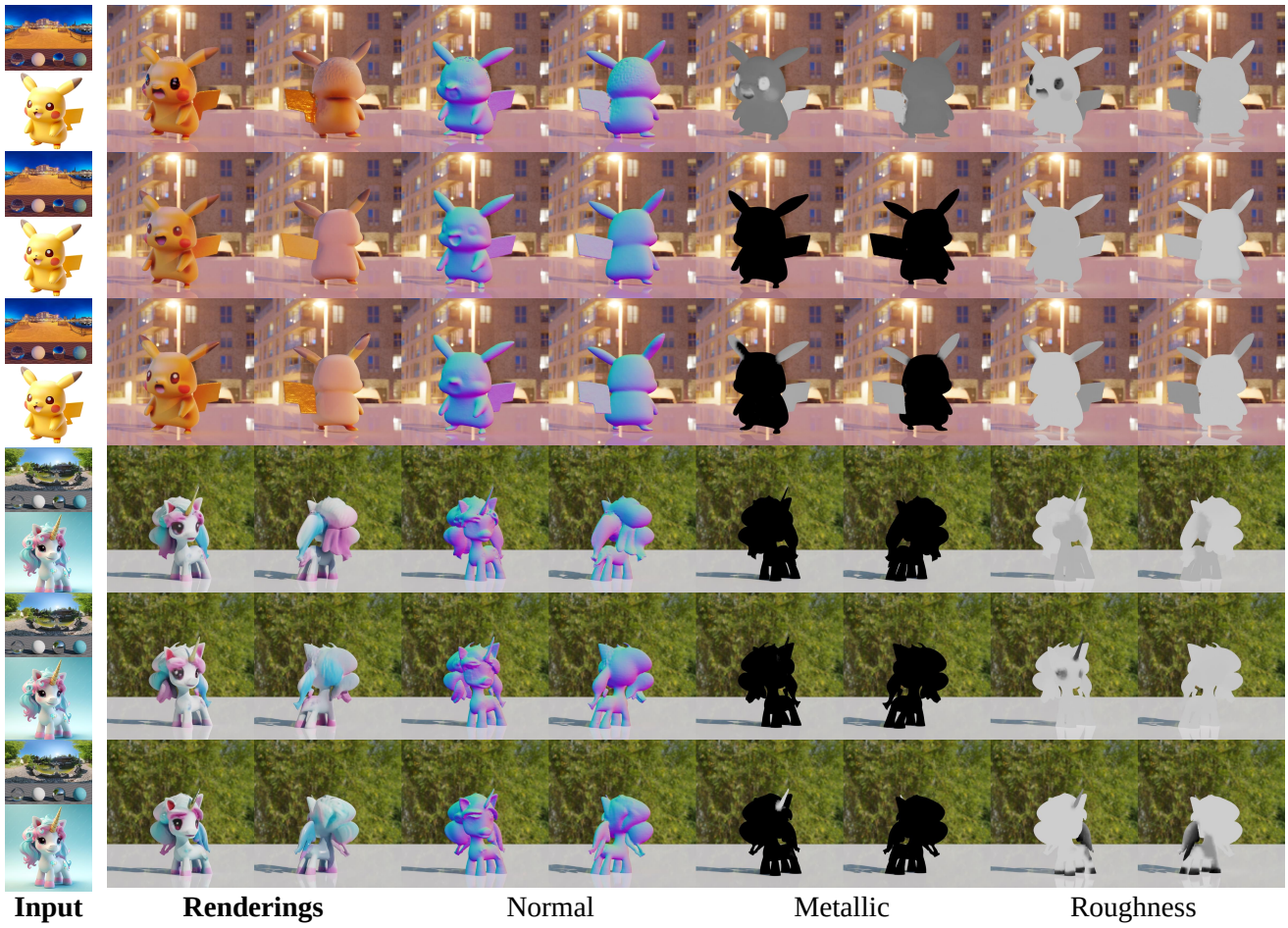


Figure 7. **Sampling diversity.** Given the same input image, 3DTopia-XL can generate diverse 3D assets by varying random seeds only. Zoom in for diverse shapes and spatially varied PBR materials.


Figure 8. Due to native 3D representation, 3DTopia-XL can generate well-defined and diverse **inner structures** from images / texts.

we compare our PBR mesh extraction with different design alternatives including:

- **Vert Coloring:** a baseline that uses vertex coloring as many prior works;
- **No Material:** a baseline that does not pack PBR materials in the GLB mesh during the conversion;
- **Low-res UV:** a baseline that unwraps textures in a low-resolution UV space ($256 \times 256$);
- **No Inpainting:** a baseline that does not perform texture inpainting based on UV adjacency.

The results are presented in Figure 11. As clearly shown in the top two rows, neither vertex coloring nor ignoring material can produce vivid reflectance under natural illumination. This is due to the fact that both two modes cannot

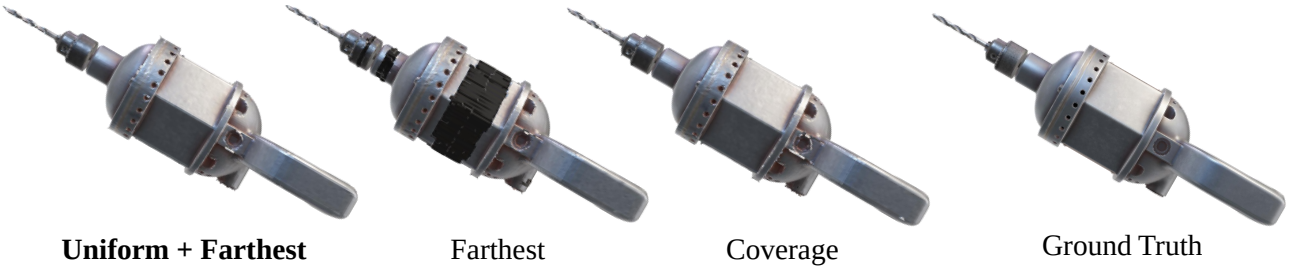


Figure 9. **The impact of different initialization strategies for mesh to PrimX algorithm (Alg. 1).**

Table 4. **Quantitative evaluations of different initialization strategies for mesh to PrimX.**

| Solution | PSNR-$F_{\mathcal{S}}^{\text{SDF}}$ ↑ | PSNR-$F_{\mathcal{S}}^{\text{RGB}}$ ↑ | PSNR-$F_{\mathcal{S}}^{\text{Mat}}$ ↑ |
|---|---|---|---|
| **Uniform + Farthest** | 72.12 | 26.26 | 21.65 |
| Farthest | 56.86 | 14.30 | 10.16 |
| Coverage | 71.38 | 26.06 | 21.41 |

support PBR materials modeling. Low-resolution UV space introduces texture noises as well as material noises. In contrast, our high-resolution UV space ($1024 \times 1024$) leads to high-quality texture representation. The baseline without UV inpainting shows severe zig-zag artifacts for the texture, which result from the empty region of UV space not properly inpainted. The above comparisons further justify the ingredients in Alg. 2 including 1) using UV texturing, 2) using high-resolution UV space, and 3) doing UV inpainting based on the vertices and faces adjacencies.

#### A.2.10. PrimX can Learn from Both 2D and 3D

Recall the three design principles for 3D representation in the context of high-quality large-scale 3D generative models: **1) Parameter-efficient**: provides a good trade-off between approximation error and parameter count; **2) Rapidly tensorizable**: can be efficiently transformed into a tensor, which facilitates generative modeling with modern neural architectures; **3) Differentiably renderable**: compatible with differentiable renderer, enabling learning from both 3D and 2D data.

PrimX is the representation that satisfies all the above criteria. We have extensively introduced how to learn from 3D dataset [5] for PrimXin Alg. 1 and Alg. 2. Here, we further demonstrate the great potential of PrimX to leverage knowledge from 2D images. Thanks to our explicit design that maintains the differentiability of the rendering process of PrimX, this tensorial 3D representation can be efficiently rasterized into 2D images given a perspective camera view. By back-propagating the gradient of image reconstruction loss to the payload in PrimX, we can improve the texture quality represented in PrimX, as shown in Figure 12. This further demonstrates that PrimX also has its potential in sparse-view reconstruction models similar to LGM [26] and LRM [9]. By replacing the underlying 3D representation as PrimX, we can also unlock a reconstruction-based model that can learn from both 2D and 3D data.

#### A.2.11. More Results

We present more image-conditioned and text-conditioned generation results in Figure 1 and Figure 2 respectively.

### A.3. Implementation Details

#### A.3.1. Data Standardization

**Datasets.** The scale and quality of 3D data determine the quality and effectiveness of 3D generative models at scales. We filter out low-quality meshes, such as fragmented shapes and large-scale scenes, resulting in a refined collection of 256k objects from Objaverse [5].

**Standardization Pipeline.** Computing PrimX on large-scale datasets involves two critical steps: **1)** Instantiation of sampling functions $\{F_{\mathcal{S}}^{\text{SDF}}, F_{\mathcal{S}}^{\text{RGB}}, F_{\mathcal{S}}^{\text{Mat}}\}$ from a GLB file and **2)** Execution of the fitting algorithm in Sec. 3.1.2 of the main paper, *i.e.* Alg. 1. Given the massive amount of meshes from diverse sources in Objaverse, there are challenges for properly instantiating the sampling functions in a universal way such as fragmented meshes, non-watertight shapes, and inconsistent UVs. We develop a unified data loading pipeline that standardizes the objects across different textured mesh definitions (vertex color, UV map, part-wise material, etc.). Our standardized procedure starts with loading the GLB file as a connected graph. We filter out subcomponents that have less than 3 face adjacency which typically represent isolated planes or grounds. After that, all mesh subcomponents are globally normalized to the unit cube $[-1, 1]$ given one unique global bounding box. Then, we instantiate geometric sampling functions


Figure 10. **Ablation studies on different VAE designs for Primitive Patch Compression.** For patch-wise spatial compression, we additionally compare our method with two alternatives: 1) PCA and 2) DCT given various compression rates.

for each mesh subcomponent for SDF, texture, and material values. For the model that lack PBR material, we will assign a default diffuse material according to Blender's principle BSDF node.

**PrimX Hyperparameters.** To get a tradeoff between computational complexity and approximation error, we choose our PrimX to have $N = 2048$ primitives where each primitive's payload has a resolution of $a = 8$. It indicates that the sequence length of our primitive diffusion Transformer is also 2048 where each token has a dimension of $d = 3 + 1 + (a/2)^3 = 68$. For the rapid finetuning stage for computing PrimX, we sample 500k points from the target mesh, where 300k points are sampled on the surface and 200k points are sampled with a standard deviation of 0.01 near the surface. The finetuning stage is run for 2k iterations with a batch size of 16k points using an Adam [13] optimizer at a learning rate of $1 \times 10^{-4}$.

### A.3.2. Condition Signals

**Conditioners.** The conditional generation formulation in Sec. 3.3 of the main paper is compatible with most modalities. In this paper, we mainly explored conditional generation on two modalities, images and texts. For image-conditioned models, we leverage pretrained DINOv2 model [18], specifically "DINOv2-ViT-B/14"[2], to extract visual tokens from input images (at a resolution of $518 \times 518$) and take it as the input condition $\mathbf{c}$. For text-conditioned models, we leverage the text encoder of the pretrained image-language model [21], namely "CLIP-ViT-L/14"[3], to extract language tokens from input texts.

**Images.** Thanks to our high-quality representation PrimX and its capability for efficient rendering, we do not need to undergo the complex and expensive rendering process like other works [9], which renders all raw meshes into 2D images for training. Instead, we opt to use the front-view image rendered by Eq. 7 which is **1)** efficient enough to compute on-the-fly, and **2)** consistent with the underlying representation compared with the rendering from the raw mesh.

[2] https://github.com/facebookresearch/dinov2
[3] https://github.com/mlfoundations/open_clip


Figure 11. **Ablation study on PrimX to mesh algorithm (Alg. 2).** Please check Sec. A.2.9 for more details of our experimental setup.

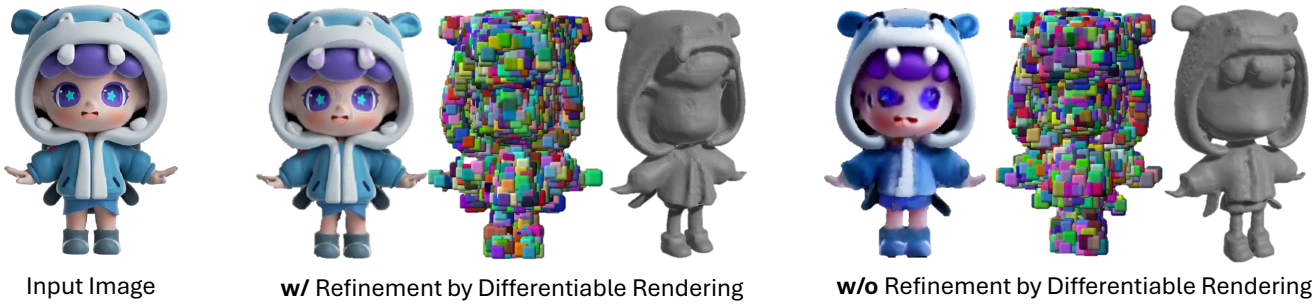


Figure 12. **The capability of PrimX to learn from 2D data.** Thanks to being differentiable renderable, PrimX can also learn from 2D images to further refine the results. This also implies its great potential to train on heterogeneous data from both 2D and 3D collections.

**Text Captions.** We use 200,000 samples from Objaverse to generate text captions. For each object, six different views are rendered against a white background. We then use GPT-4V to generate keywords based on these images, focusing on aspects such as geometry, texture, and style. While we pre-define certain keywords for each aspect, the model is also encouraged to generate more context-specific keywords. Once the keywords are obtained, GPT-4 is employed to summarize them into a single sentence, beginning with 'A 3D model of...'. These text captions are subsequently prepared as input conditions.

### A.3.3. Model Details

**Architecture.** We train the latent primitive diffusion model $g_\Phi$ using a Transformer-based architecture [19] for scalability. Our final model (Eq. 11 in the main paper) is built with 28 layers with 16-head attentions and 1152 hidden dimensions, leading to a total number of $\sim$1B parameters. Moreover, we employ the pre-normalization scheme [31] for training stability. For noise scheduling, we

**Algorithm 1:** Computing PrimX from a Textured Mesh (GLB format)

**Input :** GLB mesh $F_{\mathcal{S}}$, number of primitives $N$, voxel resolution $a$, number of candidates $\hat{N}$

▷ *Initialization*

$F_{\mathcal{S}} \leftarrow (F_{\mathcal{S}}^{\text{SDF}} \oplus F_{\mathcal{S}}^{\text{RGB}} \oplus F_{\mathcal{S}}^{\text{Mat}})$ ▷ parse volumetric sampling functions

$\{\hat{\mathbf{t}}_k\}_{k\in[\hat{N}]} \leftarrow$ uniform random sampling of $\partial\mathcal{S}$

$\{\mathbf{t}_k\}_{k\in[N]} \leftarrow$ farthest point sampling of $\{\hat{\mathbf{t}}_k\}_{k\in[\hat{N}]}$

**for** $i \leftarrow 1$ **to** $N$ **do**

- $s_i \leftarrow$ L2 distance to its nearest neighbors in $\{\mathbf{t}_k\}_{k\in[N]}$
- $\boldsymbol{X}_i^{\text{SDF}} \leftarrow F_{\mathcal{S}}^{\text{SDF}}(\mathbf{t}_i + s_i\boldsymbol{I})$ ▷ $\boldsymbol{I}$ is the local voxel grid
- $\mathbf{t}_i^{\text{uv}} \leftarrow$ UV and barycentric coordinates of the nearest face for $(\mathbf{t}_i + s_i\boldsymbol{I})$
- $\boldsymbol{X}_i^{\text{RGB}} \leftarrow F_{\mathcal{S}}^{\text{RGB}}(\mathbf{t}_i^{\text{uv}})$
- $\boldsymbol{X}_i^{\text{Mat}} \leftarrow F_{\mathcal{S}}^{\text{Mat}}(\mathbf{t}_i^{\text{uv}})$
- $\boldsymbol{X}_i \leftarrow (\boldsymbol{X}_i^{\text{SDF}} \oplus \boldsymbol{X}_i^{\text{RGB}} \oplus \boldsymbol{X}_i^{\text{Mat}})$ ▷ $\oplus$ denotes concatenation
- $\mathcal{V}_i \leftarrow \{\mathbf{t}_i, s_i, \boldsymbol{X}_i\}$

$\mathcal{V} \leftarrow \{\mathcal{V}_k\}_{k\in[N]}$

▷ *Rapid Finetuning*

**while** *not converged* **do**

- $\{\mathbf{x}_i\}_{i\in[B]} \leftarrow$ random sampling of $\mathcal{U}(\partial\mathcal{S}, \delta)$ with a batch size of $B$
- Take a gradient descent step with $\nabla_{\mathcal{V}}\mathcal{L}(\mathbf{x}; \mathcal{V})$ ▷ Eq. 9 in the main paper

**Output:** $\mathcal{V}$

**Algorithm 2:** Extracting a Textured Mesh (GLB format) from PrimX

**Input :** PrimX $\mathcal{V} = \{\mathbf{t}_k, s_k, \boldsymbol{X}_k\}_{k\in[N]}$, Marching Cubes resolution $A$, chunk size $B$

$\{F_{\mathcal{V}}^{\text{SDF}}, F_{\mathcal{V}}^{\text{RGB}}, F_{\mathcal{V}}^{\text{Mat}}\} \leftarrow F_{\mathcal{V}}$

▷ *Shape Extraction*

$\{\mathbf{x}_i\}_{i\in[A^3]} \leftarrow$ Initialize a unit cube with a resolution of $A \times A \times A$

**for** $i \leftarrow 1$ **to** $A^3$ **do**

- **if** $\min_k ||\mathbf{x}_i - \{\mathbf{t}_k\}_{k\in[N]}||_2 > s_k$ **then**
  - $F_{\mathcal{S}}^{\text{SDF}}(\mathbf{x}_i) \leftarrow \min_k ||\mathbf{x}_i - \{\mathbf{t}_k\}_{k\in[N]}||_2 \cdot \text{sign}(\boldsymbol{X}_k^{\text{SDF}})$ ▷ No query of PrimX
- **else**
  - $F_{\mathcal{S}}^{\text{SDF}}(\mathbf{x}_i) \leftarrow F_{\mathcal{V}}^{\text{SDF}}(\mathbf{x}_i)$ ▷ Run in parallel with a chunk size $B$ in practice

$\{\mathbb{V}, \mathbb{F}\} \leftarrow$ Marching Cubes on the zero level set of $\{F_{\mathcal{S}}^{\text{SDF}}(\mathbf{x}_i)\}_{i\in[A^3]}$

▷ *Texture and Material Extraction*

Empty texture maps $(F_{\mathcal{S}}^{\text{RGB}}, F_{\mathcal{S}}^{\text{Mat}})$ and UV Mapping $\leftarrow$ UV unwrapping on $\{\mathbb{V}, \mathbb{F}\}$

$\{\mathbf{x}_i^{\text{uv}}\} \leftarrow$ Get validate sampling points in 3D with a rasterizer

$F_{\mathcal{S}}^{\text{RGB}}(\mathbf{x}_i^{\text{uv}}) \leftarrow F_{\mathcal{V}}^{\text{RGB}}(\mathbf{x}_i^{\text{uv}})$

$F_{\mathcal{S}}^{\text{Mat}}(\mathbf{x}_i^{\text{uv}}) \leftarrow F_{\mathcal{V}}^{\text{Mat}}(\mathbf{x}_i^{\text{uv}})$

$(F_{\mathcal{S}}^{\text{RGB}}, F_{\mathcal{S}}^{\text{Mat}}) \leftarrow$ inpainting with nearest neighbors based on UV mapping adjacency

$\mathcal{S} \leftarrow \{\mathbb{V}, \mathbb{F}, F_{\mathcal{S}}^{\text{RGB}}, F_{\mathcal{S}}^{\text{Mat}}, \text{UV Mapping}\}$ ▷ Packed in GLB format

**Output:** $\mathcal{S}$

use discrete 1,000 noise steps with a cosine scheduler during training. We opt for "v-prediction" [23] with Classifier-Free Guidance (CFG) [8] as the training objective for better conditional generation quality and faster convergence.

**Channel-wise Normalization.** Most importantly, given the distribution gap between the 3D coordinate $\mathbf{t}$ and the latent $E(\boldsymbol{X})$, one may carefully deal with the normalization of the input data to the diffusion model. Recall our diffusion target is a hybrid tensor $\mathcal{V} = \{\mathbf{t}, s, E(\boldsymbol{X})\}$, where $E(\boldsymbol{X})$ is the 3D latent in the KL-regularized VAE that is close to

a Gaussian distribution. However, the 3D coordinate $\mathbf{t}$ is not normally distributed in the 3D space. This inter-channel distribution gap within the diffusion target will lead to sub-optimal convergence if the data is globally normalized by a scalar (which is the common practice in 2D diffusion models[4]). Intuitively, our latent primitive diffusion model aims to solve a hybrid problem of point diffusion [17] and latent diffusion [22] simultaneously. To bridge this gap, we propose to normalize the input data in a channel-wise manner. Specifically, we trace channel-wise statistics (mean and standard deviation) over 50k random samples from the dataset. During the training phase, we keep them as constant normalizing factors and apply them to the input of the latent primitive diffusion model.

**Training.** We train $g_\Phi$ with a batch size of 1024 using an AdamW [16] optimizer. The learning rate is set to $1 \times 10^{-4}$ with a cosine learning rate warmup for 3k iterations. The probability of condition dropout for CFG is set to $p_0 = 0.1$. During training, we apply EMA (Exponential Moving Average) on the model's weight with a decay of 0.9999 for better training stability. The image-conditioned model is trained on 16 nodes of 8 A100 GPUs for 350k iterations, which takes around 14 days to converge. The text-conditioned model is trained on 16 nodes of 8 A100 GPUs for 200k iterations, which takes around 5 days to converge.

**VAE.** The 3D VAE for patch-wise primitive compression is built with 3D convolutional layers. We train the VAE on a subset of the entire dataset with 98k samples, finding it generalizes well on unseen data. The training takes 60k iterations with a batch size of 256 using an Adam [13] optimizer with a learning rate of $1 \times 10^{-4}$. Note that, this batch size indicates the total number of PrimX samples per iteration. As our VAE operates on each primitive independently, the actual batch size would be $N \times 256$. We set the weight for KL regularization to $\lambda_{\text{kl}} = 5 \times 10^{-4}$. The training is distributed on 8 nodes of 8 A100 GPUs, which takes about 18 hours.

**Inference.** By default, we evaluate our model with a 25-step DDIM [25] sampler and CFG scale at 6. We find the optimal range of the DDIM sampling steps is $25 \sim 100$ while the CFG scale is $4 \sim 10$. The inference can be efficiently done on a single A100 GPU within 5 seconds.

#### A.3.4. Reversible Conversion between PrimX and Mesh

**Mesh to PrimX.** As introduced in the main paper (Sec. 3.1.2), we leverage a two-stage strategy to compute PrimX from a textured mesh. Given a textured mesh $F_S$ that contains the shape, albedo, and material information, we convert it into PrimX with $N$ primitives via a good initialization followed by a rapid finetuning. Here, we introduce more details of this procedure in Algorithm 1. Our implementation to instantiate the volumetric sampling function of SDF that works for non-watertight mesh is derived from cuBVH[5].

**PrimX to Mesh.** As introduced in the main paper (Sec. 3.1.1), PrimX can be inversely converted back to a textured mesh in GLB format with minimal loss of information. The key is to utilize a high-resolution UV space for texturing instead of vertex coloring. We specify the details of this procedure in Algorithm 2, where we use xatlas[6] for UV unwrapping, nvdiffrast[7] for mesh-based rasterizer, and mcubes[8] for Marching Cubes [15].

[4] https://github.com/huggingface/diffusers/issues/437

[5] https://github.com/ashawkey/cubvh

[6] https://github.com/jpcy/xatlas

[7] https://github.com/NVlabs/nvdiffrast

[8] https://github.com/pmneila/PyMCubes

## References


[1] Mikołaj Bińkowski, Dougal J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *International Conference on Learning Representations*, 2018. 3

[2] Zoya Bylinskii, Laura Herman, Aaron Hertzmann, Stefanie Hutka, and Yile Zhang. Towards better user studies in computer graphics and vision. *arXiv preprint arXiv:2206.11461*, 2022. 3

[3] Zhaoxi Chen, Fangzhou Hong, Haiyi Mei, Guangcong Wang, Lei Yang, and Ziwei Liu. Primdiffusion: Volumetric primitives diffusion for 3d human generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 2

[4] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 4, 5

[5] 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 *CVPR*, pages 13142–13153, 2023. 1, 2, 3, 9

[6] 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, 2022. 1, 2, 3

[7] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a nash equilibrium. *CoRR*, abs/1706.08500, 2017. 3

[8] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 12

[9] 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. 5, 9, 10

[10] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023. 2, 3

[11] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *Proc. CVPR*, 2020. 2

[12] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ToG*, 42(4):1–14, 2023. 2

[13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 10, 13

[14] Yushi Lan, Fangzhou Hong, Shuai Yang, Shangchen Zhou, Xuyi Meng, Bo Dai, Xingang Pan, and Chen Change Loy. Ln3diff: Scalable latent neural fields diffusion for speedy 3d generation. *arXiv preprint arXiv:2403.12019*, 2024. 2, 3

[15] 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. 13

[16] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 13

[17] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022. 13

[18] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 6, 7, 10

[19] William Peebles and Saining Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022. 3, 11

[20] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, 2017. 2, 3

[21] 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 *ICML*, pages 8748–8763. PMLR, 2021. 10

[22] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pages 10684–10695, 2022. 13

[23] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022. 12

[24] 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. 3

[25] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 13

[26] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. *arXiv preprint arXiv:2402.05054*, 2024. 2, 3, 9

[27] Gregory K. Wallace. The jpeg still picture compression standard. *Commun. ACM*, 34(4):30–44, 1991. 7

[28] Peng Wang and Yichun Shi. Imagedream: Image-prompt multi-view diffusion for 3d generation. *arXiv preprint arXiv:2312.02201*, 2023. 3

[29] Zhengyi Wang, Yikai Wang, Yifei Chen, Chendong Xiang, Shuo Chen, Dajiang Yu, Chongxuan Li, Hang Su, and Jun Zhu. Crm: Single image to 3d textured mesh with convolutional reconstruction model. *arXiv preprint arXiv:2403.05034*, 2024. 2, 3

[30] Kailu Wu, Fangfu Liu, Zhihan Cai, Runjie Yan, Hanyang Wang, Yating Hu, Yueqi Duan, and Kaisheng Ma. Unique3d: High-quality and efficient 3d mesh generation from a single image, 2024. 3

[31] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning*, pages 10524–10533. PMLR, 2020. 11

[32] Jiale Xu, Weihao Cheng, Yiming Gao, Xintao Wang, Shenghua Gao, and Ying Shan. Instantmesh: Efficient 3d mesh generation from a single image with sparse-view large reconstruction models. *arXiv preprint arXiv:2404.07191*, 2024. 2, 3

[33] Lior Yariv, Omri Puny, Natalia Neverova, Oran Gafni, and Yaron Lipman. Mosaic-sdf for 3d generative models. *arXiv preprint arXiv:2312.09222*, 2023. 2, 3, 7

[34] 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. 1, 3

[35] Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. Clay: A controllable large-scale generative model for creating high-quality 3d assets. *arXiv preprint arXiv:2406.13897*, 2024. 1, 3