TEXGen: a Generative Diffusion Model for Mesh Textures

XIN YU, The University of Hong Kong, Hong Kong
ZE YUAN, Beihang University, China
YUAN-CHEN GUO, VAST, China
YING-TIAN LIU, Tsinghua University, China
JIANHUI LIU, The University of Hong Kong, Hong Kong
YANGGUANG LI, VAST, China
YAN-PEI CAO, VAST, China
DING LIANG, VAST, China
XIAOJUAN QI[*], The University of Hong Kong, Hong Kong

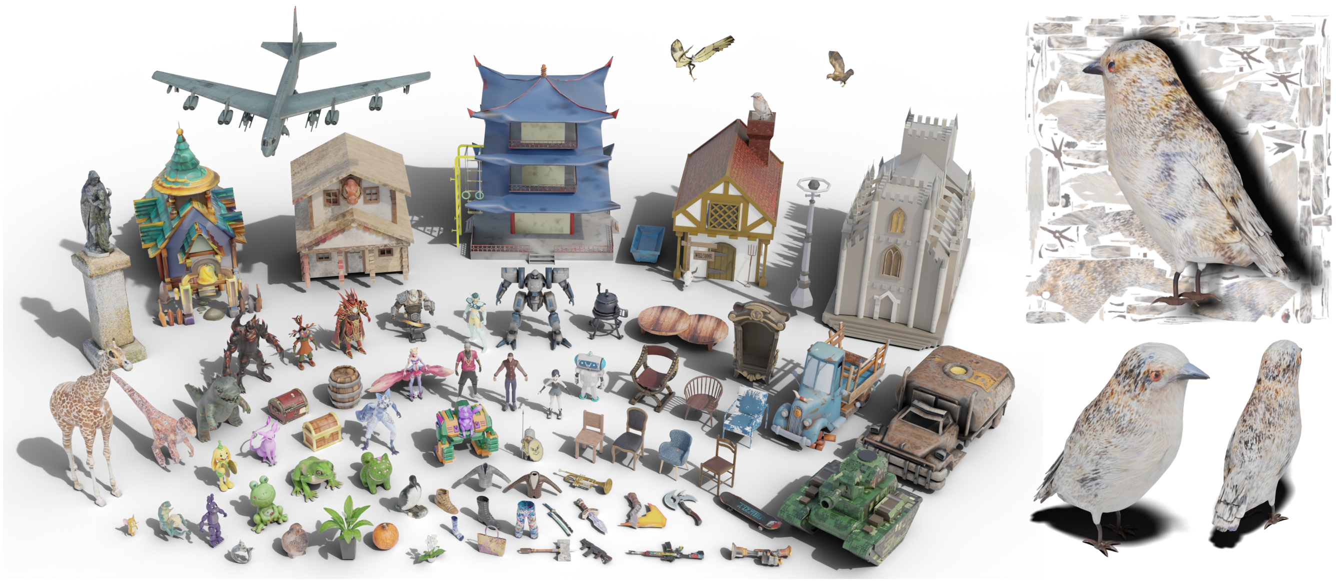


Fig. 1. **3D meshes with textures generated by our method.** We show a gallery of 3D meshes with textures generated by our method (left) and the texture map and multi-view renderings of the bird model (right). Our approach models the distribution of mesh textures at high resolution, generating high-quality textures from text and image prompts, more multi-view renderings are shown in fig. 5.

While high-quality texture maps are essential for realistic 3D asset rendering, few studies have explored learning directly in the texture space, especially on large-scale datasets. In this work, we depart from the conventional approach of relying on pre-trained 2D diffusion models for test-time optimization of 3D textures. Instead, we focus on the fundamental problem of learning in the UV texture space itself. For the first time, we train a large diffusion model capable of directly generating high-resolution texture maps in a feed-forward manner. To facilitate efficient learning in high-resolution UV spaces, we propose a scalable network architecture that interleaves convolutions on UV maps with attention layers on point clouds. Leveraging this architectural design, we train a 700 million parameter diffusion model that can generate UV texture maps guided by text prompts and single-view images. Once trained, our model naturally supports various extended applications, including text-guided texture inpainting, sparse-view texture completion, and text-driven texture synthesis. The code is available at https://github.com/anonymous4coderelease/TEXGen.

CCS Concepts: • **Computing methodologies** → **Mesh models**; **Shape analysis**; **Neural networks**.

[*]Corresponding author.

Authors' addresses: Xin Yu, The University of Hong Kong, Hong Kong, yuxin27g@gmail.com; Ze Yuan, Beihang University, China, yuanze1024@buaa.edu.cn; Yuan-Chen Guo, VAST, China, imbennyguo@gmail.com; Ying-Tian Liu, Tsinghua University, China, liuyingt23@mails.tsinghua.edu.cn; Jianhui Liu, The University of Hong Kong, Hong Kong, jhliu0212@gmail.com; Yangguang Li, VAST, China, liyangguang256@gmail.com; Yan-Pei Cao, VAST, China, caoyanpei@gmail.com; Ding Liang, VAST, China, liangding1990@163.com; Xiaojuan Qi, The University of Hong Kong, Hong Kong, xjqi@eee.hku.hk.

ACM 0730-0301/2024/9-ART
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Additional Key Words and Phrases: Generative model, texture generation

ACM Reference Format:
Xin Yu, Ze Yuan, Yuan-Chen Guo, Ying-Tian Liu, Jianhui Liu, Yangguang Li, Yan-Pei Cao, Ding Liang, and Xiaojuan Qi. 2024. TEXGen: a Generative Diffusion Model for Mesh Textures. *ACM Trans. Graph.* 1, 1 (September 2024), 14 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Synthesizing textures for 3D meshes is a fundamental problem in computer graphics and vision, with numerous applications in virtual reality, game design, and animation. However, the most advanced learning-based methods [Cheng et al. 2023; Oechsle et al. 2019; Siddiqui et al. 2022; Yu et al. 2023a] are restricted to generating textures for specific categories due to scalability and data limitation. Recently, test-time optimization-based methods have emerged, which utilize pre-trained 2D diffusion models to produce image priors via score distillation sampling [Lin et al. 2023; Poole et al. 2022; Wang et al. 2023b; Yu et al. 2023b] or by synthesizing pseudo-multi-views [Chen et al. 2023b; Richardson et al. 2023; Zeng 2023]. While these methods can generate textures for a wide range of objects, they suffer from certain drawbacks, such as time-consuming per-object optimization and parameter tuning, susceptibility to the limitations of 2D priors, and poor 3D consistency in texture generation.

In recent years, there has been a surge in the development of large models across various domains, including natural language processing [Achiam et al. 2023; Touvron et al. 2023], image and video generation [Betker et al. 2023; Blattmann et al. 2023; Saharia et al. 2022], and 3D creation [Hong et al. 2023; Li et al. 2023; Tochilkin et al. 2024; Wang et al. 2023c; Xu et al. 2023; Zou et al. 2023]. These models produce high-quality results and demonstrate remarkable generalization capabilities. Their success can be primarily attributed to two key factors: (1) scalable and effective network architectures that improve performance as model size and data amount increase, and (2) large-scale datasets that facilitate generalization. In this paper, we explore the potential of building a large generative model by scaling up model size and data for generalizable and high-quality mesh texturing.

We introduce TEXGen, a large generative model for mesh texturing. Our model utilizes a UV texture map as the representation for generation, as it is scalable and preserves high-resolution details. More importantly, it enables direct supervision from ground-truth texture maps without solely relying on rendering loss [Hong et al. 2023; Li et al. 2023], making it compatible with diffusion-based training and improving the overall generative quality. Previous works such as Point-UV-Diffusion [Yu et al. 2023a] and Paint3D [Zeng 2023] have attempted to leverage diffusion models to learn the distribution of mesh textures. However, neither of these approaches achieved end-to-end training or feed-forward inference on general object datasets [Deitke et al. 2023], resulting in error accumulation and scalability issues.

To perform effective feature interaction on mesh surfaces, we propose a scalable 2D-3D hybrid network architecture that incorporates convolution operations in the 2D UV space, followed by sparse convolutions and attention layers operating in the 3D space. This simple yet effective architecture offers several key advantages: (1) by applying convolution operations in the UV space, the network effectively learns local and high-resolution details; and (2) by further elevating the computation into 3D space, the network can learn global 3D dependencies and neighborhood relationships that are disrupted by the UV parameterization process, ensuring global 3D coherency. This hybrid design allows us to use sparse features in 3D space instead of dense voxel [Chen et al. 2018] or point features [Nichol et al. 2022; Yu et al. 2023a] for manageable computations while still maintaining 3D continuity, making the architecture scalable. By stacking multiple blocks, we train a large texture diffusion model capable of directly synthesizing high-resolution textures (e.g., 1024×1024 texture maps) in a feed-forward manner guided by single-view images and text prompts. Moreover, our pre-trained model enables various applications, including text-guided texture synthesis, inpainting, and texture completion from sparse views.

To summarize, our contributions are as follows:

- We introduce a novel network architecture designed for learning high-resolution UV texture maps, wherein we build a hybrid 2D-3D denoising block for effective feature learning.
- Based on this architecture, we have trained a large diffusion model for high-resolution texture map generation. To the best of our knowledge, this is the first work capable of generating texture maps in an end-to-end manner without requiring additional stages, or test-time optimization.
- Our method achieves state-of-the-art results and serves as a foundation model supporting various training-free applications, such as text-guided texture synthesis, inpainting, and texture completion from sparse views.

2 RELATED WORK

Texture generation via 2D diffusion models. A prevalent method for texturing 3D meshes involves test-time optimization with pre-trained 2D diffusion models. Techniques such as those based on score distillation sampling [Chen et al. 2023a; Lin et al. 2023; Metzer et al. 2023; Poole et al. 2022; Wang et al. 2023b; Yeh et al. 2024; Yu et al. 2023b], synthesize textures on 3D shapes by distilling 2D diffusion priors. However, these approaches have significant drawbacks, including high computational demands and inherent artifacts like the Janus problem and unnatural color. Another line of approach [Cao et al. 2023; Ceylan et al. 2024; Chen et al. 2023b; Gao et al. 2024; Liu et al. 2023; Richardson et al. 2023; Wu et al. 2024; Zhang et al. 2024] leverages geometry-conditioned image generation and inpainting to progressively generate the textures. TEXTure [Richardson et al. 2023], for example, generates a partial texture map from one perspective view before using inpainting to complete other views. However, this method struggles with inconsistencies due to the lack of global information across views. Text2Tex [Chen et al. 2023b] introduces an automated strategy for optimized viewpoint selection to avoid manual intervention. Meanwhile, TexFusion [Cao et al. 2023] proposes to aggregate appearances from multiple viewpoints during the diffusion denoising steps, producing a more consistent and cohesive texture map. Despite the advances, these methods predominantly lack 3D awareness due to their reliance on 2D diffusion models and often require time-consuming per-instance optimization.

Texture generative models. A variety of learning-based approaches have been developed to train generative models from 3D data [Chang et al. 2015] for mesh texturing. Early methods [Oechsle et al. 2019] introduced implicit texture fields that assign colors to each point on the 3D surface. However, these methods often struggle to reproduce high-frequency details due to the continuous nature of implicit fields. Texturify [Siddiqui et al. 2022] and Mesh2Tex [Bokhovkin et al. 2023] design convolution operations tailored for mesh structures to facilitate learning directly on surfaces, which use the StyleGAN architecture [Karras et al. 2019] to predict textures for each mesh face and relies on GANs [Goodfellow et al. 2020] for training. Despite these advances, these methods are susceptible to mode collapse due to the instability of GAN training. More recent approaches, such as TUVF [Cheng et al. 2023] and Point-UV Diffusion [Yu et al. 2023a], attempt to generate UV maps directly for 3D shapes, addressing some of the aforementioned challenges. However, these methods are generally limited to category-specific objects and struggle with generalized objects. Paint3D has demonstrated the capability of handling generalized objects by fine-tuning a diffusion model for texture maps on larger-scale datasets [Deitke et al. 2023]. Nonetheless, it still requires test-time optimization to generate the initial textures, and the trained diffusion model is only capable of removing light effects and filling holes. The two-stage pipeline can lead to cumulative quality losses, often resulting in degenerated details in the final output.

Feed-forward methods for 3D generation. Recently, there has been a notable shift in the community towards training feed-forward 3D generative models using large-scale datasets. These models are designed to accept minimal input conditions and directly output 3D representations, thereby eliminating the need for per-instance optimization [Chen et al. 2023a; Lin et al. 2023; Metzer et al. 2023; Poole et al. 2022; Wang et al. 2023a,b; Yu et al. 2023b]. Notably, Large Reconstruction Model (LRM) and its variants [Hong et al. 2023; Li et al. 2023; Tochilkin et al. 2024; Xu et al. 2024; Zou et al. 2023] adopt a transformer-based architecture to infer 3D shapes from single or sparse-view inputs, which show significant improvements on the quality and efficiency of feed-forward 3D reconstruction. However, these methods often result in over-smoothed appearances, especially in areas not visible in the input views, and lack the capability to produce varied outcomes by design. Furthermore, adapting these models for texture generation given the 3D geometry poses significant challenges, as they typically manage feature interactions in a coarse-grained 3D space rather than directly on surfaces.

3 OVERVIEW

Given a 3D mesh S, our objective is to develop a generative model capable of producing high-quality textures for 3D surfaces based on user-defined conditions such as images or text prompts, as illustrated in fig. 3 (a). The modeling comprises the following principal steps:

(i) Data representations. We use UV texture maps as the mesh texture representation, which is compact and suitable for diffusion training. We discuss its characteristics in section 4.1 which motivate us to develop a new network architecture in section 4.2.

(ii) Model construction and learning. We develop a novel hybrid 2D-3D network structure that effectively handles the unique characteristics of texture maps (section 4.2). We then train a diffusion model [Ho et al. 2020] to generate high-resolution texture maps for a given mesh based on a single-view image and a text description (section 4.3).

(iii) Inference. After training is done, our model can start from a noise image and iteratively denoise it to generate high-resolution texture maps. Additionally, our model supports various training-free extensions, such as text-guided texture synthesis, texture inpainting, and texture completion from sparse views (section 4.4).

4 METHOD

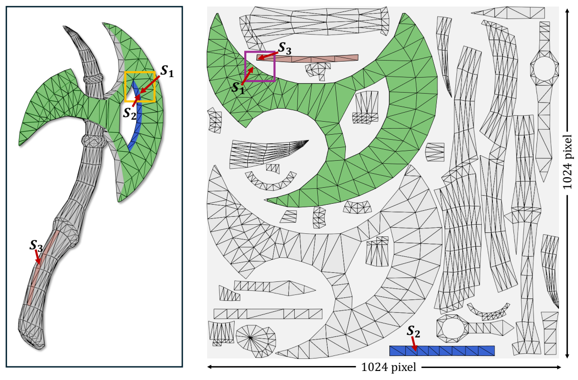
4.1 Representation for Texture Synthesis

A surface can be fundamentally viewed as a two-dimensional signal embedded within a three-dimensional space. Consequently, a traditional technique in graphics for processing mesh structures is UV mapping, which flattens the 3D structure into a compact 2D representation (refer to fig. 2). This transformation allows 3D attributes, such as textures, to be reorganized and represented on a 2D plane. The 2D UV space effectively captures neighborhood dependencies within individual islands, enhancing computational efficiency for texture generation [Yu et al. 2023a] thanks to its grid structure. Additionally, the explicit nature of the texture map facilitates direct supervision, making it well-suited for integration with diffusion models.

The advantages outlined above motivate our adoption of a 2D UV texture map as the representation for texturing 3D meshes. However, despite its merits, this approach inevitably loses the global-level 3D consistency among different islands due to the fragmentation inherent in UV mapping. As illustrated in fig. 2, islands S_1 and S_2 are contiguous on the 3D surface but are positioned far apart on the UV map. Conversely, S_1 and S_3, which are adjacent on the UV map, do not share a physical connection on the surface. This fragmentation can lead to inaccurate feature extraction in conventional image-based models. To address this issue, we propose a novel model that synergizes the strengths of the 2D UV space—enabling high-resolution and detailed feature learning—with the incorporation of 3D points to maintain global consistency and continuities. These components interleave and refine representations, facilitating effective learning for generating high-resolution 2D texture maps. Further details will be in section 4.2.

4.2 Model Construction

Utilizing 2D texture representations, we can train a diffusion model that conducts iterative denoising to generate a high-quality 2D texture map, given a specific condition, such as a posed single image or a text prompt. The core of our model is a hybrid 2D-3D network that learns features in both 2D and 3D spaces (see fig. 3). Unlike unconditional generation, our work prioritizes conditional generation, particularly conditioning on textual and visual inputs. Text prompts provide an intuitive interface for users to specify desired attributes in the generated content, making the model more accessible and responsive to user intentions. On the other hand, conditioning on images offers precise control over the generation process by capturing pixel-level details that text alone may overlook, thus offering stronger guidance for diffusion models. Moreover, a single image



(a) A triangle mesh (b) UV map

Fig. 2. **An illustration of (a) a mesh with its (b) UV map.** Three islands S_1, S_2 and S_3 are shown both on the mesh surface and its flattened UV map, where continuous islands S_1 and S_2 are positioned far apart on the UV map while disconnected islands S_1 and S_3 show closer distance on the UV map.

with rich textures can serve as a valuable prior in the diffusion process, facilitating more effective learning. Since it is feasible to generate an image from text alone using text-to-image models [Rombach et al. 2022; Zhang et al. 2023], we choose to condition on both text and images for training. During inference, the model remains flexible, allowing for the inclusion or omission of image data based on its availability to the user (see section 4.4).

Network. As depicted in fig. 3, our training pipeline utilizes a diffusion-based approach. At each denoising step, our network processes multiple inputs: the noised texture map x_t, position map x_{pos}, mask map x_{mask}, a single image I, text prompt c, and timestep t, to guide the removal of noise from x_t. The integration of the image I into the network occurs in two distinct ways: **(1) Projection of image pixels:** The image pixels are projected back onto the surface to derive a partial texture map x_I, which serves as an additional input. **(2) Global embeddings extraction:** Using an image encoder from CLIP [Radford et al. 2021] and a text encoder, we extract global image and text embeddings, respectively. A learnable timestep embedding accommodates different values of t. These embeddings are processed through separate MLPs and subsequently combined to form the global condition embedding y. This embedding modulates features within the network to incorporate condition-specific information, similar to [Peebles and Xie 2023]. The network predicts the velocity v_t [Salimans and Ho 2022], which can be equivalently transformed into a prediction of noise ϵ or a prediction of x_0. Similar to the denoising network described in [Ho et al. 2020], our architecture is based on the UNet framework [Ronneberger et al. 2015]. However, we uniquely enhance it by incorporating hybrid 2D-3D blocks at each stage. This adaptation enables our network to adeptly manage the distinct characteristics of texture maps.

Hybrid 2D-3D block. The key to our design is the hybrid 2D-3D block, which facilitates efficient feature learning for 2D texture map generation. As shown in fig. 3 (b), our hybrid block comprises a UV head and several 3D point-cloud blocks. An input UV feature f_{in} is first processed through a 2D convolution block (see fig. 3 (c)) to extract local features in the UV space. 2D convolutions are computationally more efficient compared to 3D convolutions or point cloud KNN searches for establishing neighbors and weighting, making it more scalable to higher resolution. Furthermore, within an island, 2D convolutions ensure that the aggregation of adjacent features is based on surface neighborhoods rather than volumetric neighborhoods, where geodesic distances can be larger. Thus, this step efficiently ensures the preservation of high-resolution information.

To establish 3D connections among islands in UV space, we employ rasterization to remap the output UV features $f_{\text{out}}^{\text{uv}}$ back into the 3D space, thus reorganizing these UV features into 3D point cloud features $f_{\text{in}}^{\text{point}}$. The primary objective in the 3D space is to acquire 3D neighborhood relationships and global structural features to improve 3D consistency rather than extracting high-resolution detail features. Consequently, we employ relatively sparse features and design an efficient module to ensure scalability. A brief illustration is shown in fig. 4. The key components are detailed as follows:

- **Serialized attention.** For the input dense point features $f_{\text{in}}^{\text{point}}$, we adopt grid-pooling [Wu et al. 2022] to sparsify the number of points and obtain $f_{\text{in}}^{\text{sp}}$. The pooled features are then treated as tokens and processed by point attention layers for learning. To boost the efficiency, we utilize Serialized Attention [Wu et al. 2023], which facilitates efficient patch-based attention. Specifically, the point features are partitioned into different groups based on their sterilized codes, defined by space-filling curves, such as the z-order curve [Morton 1966] and the Hilbert curve [Hilbert and Hilbert 1935].
- **Position encoding.** Position encoding plays a critical role in incorporating 3D position information into our model. Traditional methods that use point coordinates as cues for position encoding [Lai et al. 2022; Yang et al. 2023] are less effective compared to conditional positional encoding [Chu et al. 2021; Wang 2023; Wu et al. 2023], which utilizes convolution layers for this purpose [Wu et al. 2023]. Initially, we implemented the xCPE [Wu et al. 2023], which integrates a sparse convolution layer directly before the attention layer. However, this approach proved to be inefficient and time-consuming as the transformer dimension increases (i.e., $d = 2048$). To address these inefficiencies, we developed a modified approach, termed sCPE, which uses a linear layer to reduce the input's channel dimension before executing the sparse convolution. Subsequently, another linear layer is used to expand the channel dimension back to its original size to match the feature dimension of the skip connection.
- **Condition modulation.** For both our 2D and 3D blocks, we utilize the global condition embedding to modulate intermediate features, enabling the injection of conditional information. Specifically, inspired by DiT [Peebles and Xie 2023], we employ MLPs to learn modulation vectors γ and β from the condition embedding y. These vectors are used to scale and shift the intermediate features across their channel dimensions, formulated as $f_{\text{mod}} = (1 + \gamma) \cdot f_{\text{in}} + \beta$. Besides, we also learn a gated scale α to scale the output feature before its fusion with the feature from the skip connection, expressed as $f_{\text{fuse}} = \alpha \cdot f_{\text{out}} + f_{\text{skip}}$.

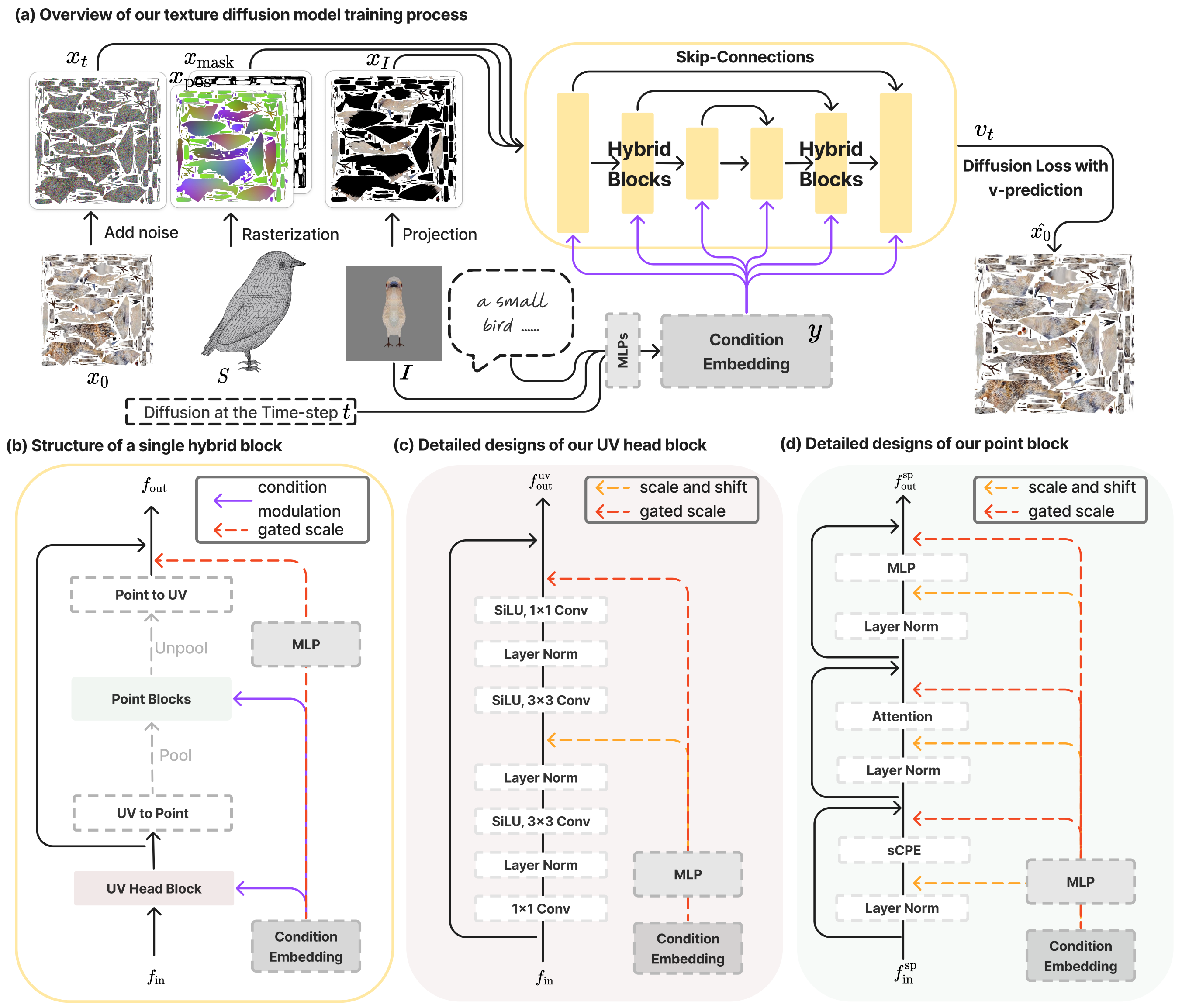


Fig. 3. **An overview of TEXGen.** (a). An overview of our training pipeline. We train a diffusion model to generate high-resolution texture maps for a given mesh S based on a single-view image I and text descriptions by learning to denoise from a noise texture map x_t. The core of our denoising network is our proposed hybrid 2D-3D block. (b). The structure of a single hybrid block. (c)-(d). The detailed designs of our UV head block and point block.

The learned sparse point features $f^{\text{sp}}_{\text{out}}$ are then scattered to dense coordinates based on grid partition, resulting in $f^{\text{point}}_{\text{out}}$. Before fusion with the skip-connected UV feature, we also learn a gated scale α^{point} from the condition embedding y to scale the point features. The final fused feature is given by: $f_{\text{out}} = f^{\text{uv}}_{\text{out}} + \alpha^{\text{point}} \cdot f^{\text{point}}_{\text{out}}$

4.3 Diffusion Learning

Given a real texture map x_0, we randomly sample a time-step t ($t \in \{0, 1, \ldots, 1000\}$) and add noise to the texture map by

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, \tag{1}$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, and $\{\bar{\alpha}_t\}$ are hyperparameters that follow a certain noise scheduler. Specifically, we use a noise scheduler from Stable Diffusion [Rombach et al. 2022] and adopt zero-terminal SNR [Lin et al. 2024] to scale the original noise scheduler such that $\bar{\alpha}_t = 0$ when $t = 1000$. This helps eliminate the gap between training and inference at the initial starting point. During training, we randomly drop text embeddings and image embeddings with a probability $p = 0.2$ so that we can utilize classifier-free guidance [Ho and Salimans 2022] during inference. For the network output x_{out}, we use v-prediction [Lin et al. 2024; Salimans and Ho 2022] to compute

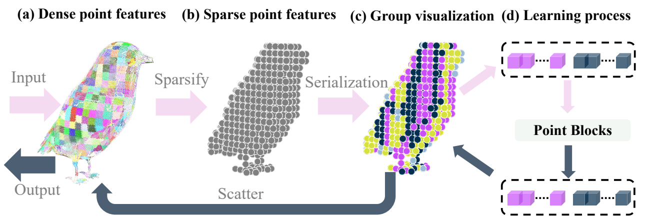


Fig. 4. **An illustration of the feature learning procedure in 3D space.** In panel (a), we start with rasterized dense point features, which we sparsify using grid-pooling to create sparse point features shown in (b). Different pools are indicated by various colors in (a). These points are then serialized to determine their order for subsequent group-based self-attention, as part of the learning process shown in (d). In (c), we visualize different groups formed based on Hilbert serialization, where each color signifies a distinct group. Finally, the processed features are scattered back to their original coordinates, providing the output dense point features.

the diffusion loss, i.e.,

$$\begin{aligned} v_t &= \sqrt{\bar{\alpha}_t}\epsilon - \sqrt{1-\bar{\alpha}_t}x_0, \\ \mathcal{L}_{\text{diff}} &= \lambda_t \left\| v_t - x_{\text{out}} \right\|_2^2, \end{aligned} \tag{2}$$

where λ_t is the soft-min-SNR weight [Crowson et al. 2024].

Notably, we obtain the predicted sample $\hat{x_0}$ from the v-prediction output and apply an LPIPS [Zhang et al. 2018] loss on multi-view renderings for extra supervision:

$$\mathcal{L}_{\text{render}} = \frac{1}{N}\sum \text{LPIPS}(\hat{I}_i, I_i), \tag{3}$$

where $\hat{I}_i$ is an image rendered from a random viewpoint using the predicted texture map $\hat{x_0}$, and I_i is the corresponding ground truth image. Our final loss is:

$$\mathcal{L} = \lambda_1\mathcal{L}_{\text{diff}} + \lambda_2\mathcal{L}_{\text{render}}, \tag{4}$$

where we set $\lambda_1 = 1$ and $\lambda_2 = 0.5$.

4.4 Texture Generation

After training, our denoising network is ready to generate high-quality texture maps for 3D meshes (see fig. 1 and fig. 5). We start by initializing a pure Gaussian noise texture map in UV space. Then, using conditional information (e.g., single-view image, text prompt), we iteratively denoise it to generate a final texture map. To accelerate inference, we use DDIM [Song et al. 2020] sampling for 30 steps. Interestingly, although we trained our model guided by a single-view image and text prompt, it can be generalized to other scenarios and applications during testing.

Text to texture generation. If only a text prompt is provided, we can arbitrarily choose a mesh viewpoint, render a depth map, and then use ControlNet [Zhang et al. 2023] to generate a corresponding single-view image. This is also a motivation for us to train an image-conditioned model instead of a text-conditioned model since an image can be easily obtained from text, whereas a text-conditioned model lacks the control capabilities provided by an image.

Texture inpainting. During training, the pixel information of the single-view image is projected back to UV space, resulting in a partial initial texture map. Our network is trained to fill in the unseen parts. We found that this capability allows the model to function as a texture inpainting model during testing. Specifically, we can take the user-provided partial texture map and mask as x_I (i.e., skipping the projection from the single-view step) and input them into the network for inpainting. For the image embedding required during testing, we set it to zero embedding, as our training included randomly dropping the image embedding, making the model robust to this situation.

Texture completion from sparse views. If the user provides a few sparse-view images, such as two images, we can effectively utilize the additional information for a generation. We simply project and fuse each image during the projection step and randomly select one image for image embedding extraction. Our model can fill textures in the occluded parts and recover the whole texture map.

5 EXPERIMENTS

We use Objaverse [Deitke et al. 2023] as our raw data source, which comprises over 800,000 3D meshes. After processing and cleaning this dataset, we extracted a total of 120,400 data pairs. Of these, 120,000 pairs are designated for training and the remaining 400 pairs are set aside for evaluation. Due to the page limit, we provide a detailed data processing method and implementation details of our model in the supplementary materials.

5.1 Main Results and Comparisons

We present our primary results, which include the textured 3D mesh conditioned on a single-view image and a text prompt, as illustrated in fig. 1. Notably, consider the example of the bird, where the texture detailing on the feathers demonstrates the model's capability to generate highly detailed textures. In fig. 5, we display the conditions and multi-view results of several examples independently. Our results demonstrate that the model can generate high-quality textures with rich local details, preserve condition information, and achieve global coherence. We compare our method with other generalizable texture generation methods, including TEXTure [Richardson et al. 2023], Text2Tex [Chen et al. 2023b], and Paint3D [Zeng 2023].

Qualitative comparisons. We conducted a qualitative analysis comparing our method with TEXTure and Text2Tex. Both these methods utilize a 2D pretrained text-to-image diffusion model for test-time optimization to texture 3D meshes. Their approach involves generating an image with a depth-conditioned diffusion model that aligns with the geometry of the current view, which is then projected onto the mesh. For areas without direct visibility, they iteratively adjust the viewpoint and employ an inpainting model to complete the texture. This cycle is repeated to produce a full texture map. For a fair comparison, we replaced their initial image with the same single-view image used in our method. As depicted in fig. 6, both TEXTure and Text2Tex encounter several challenges. In the first example, despite the conditional image containing rich texture patterns, both methods produce overly smooth textures. In the second and third examples, while more details are shown, various artifacts degrade

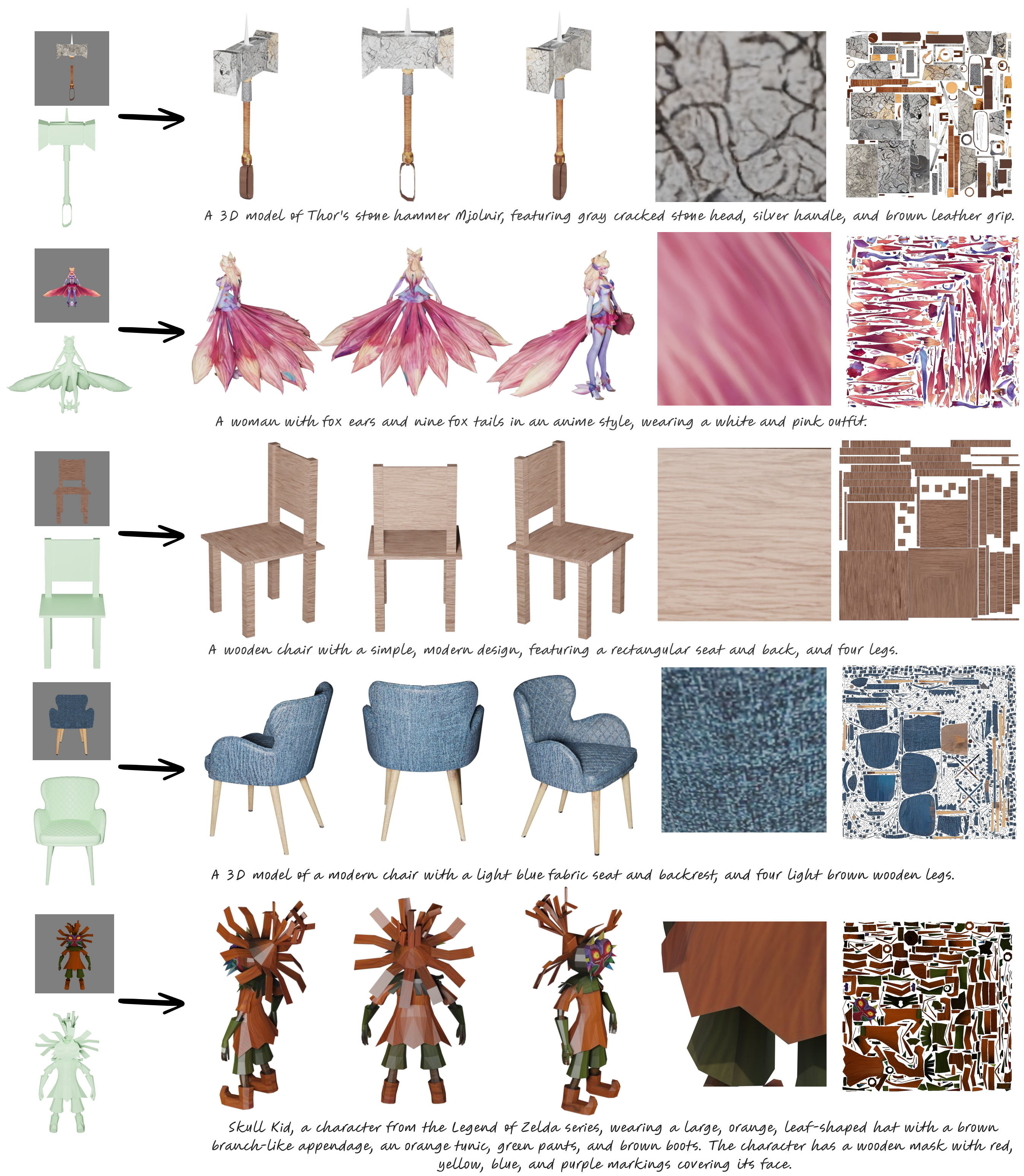

A 3D model of Thor's stone hammer Mjolnir, featuring gray cracked stone head, silver handle, and brown leather grip.
A woman with fox ears and nine fox tails in an anime style, wearing a white and pink outfit.
A wooden chair with a simple, modern design, featuring a rectangular seat and back, and four legs.
A 3D model of a modern chair with a light blue fabric seat and backrest, and four light brown wooden legs.
Skull Kid, a character from the Legend of Zelda series, wearing a large, orange, leaf-shaped hat with a brown branch-like appendage, an orange tunic, green pants, and brown boots. The character has a wooden mask with red, yellow, blue, and purple markings covering its face.

Fig. 5. **Texture generation results.** For given meshes, our method can synthesize highly detailed textures conditioned on guided single-view images and text prompts. We show three novel view images from our textured results and representative zoom-in regions from the textured mesh. The generated full texture maps are also shown.

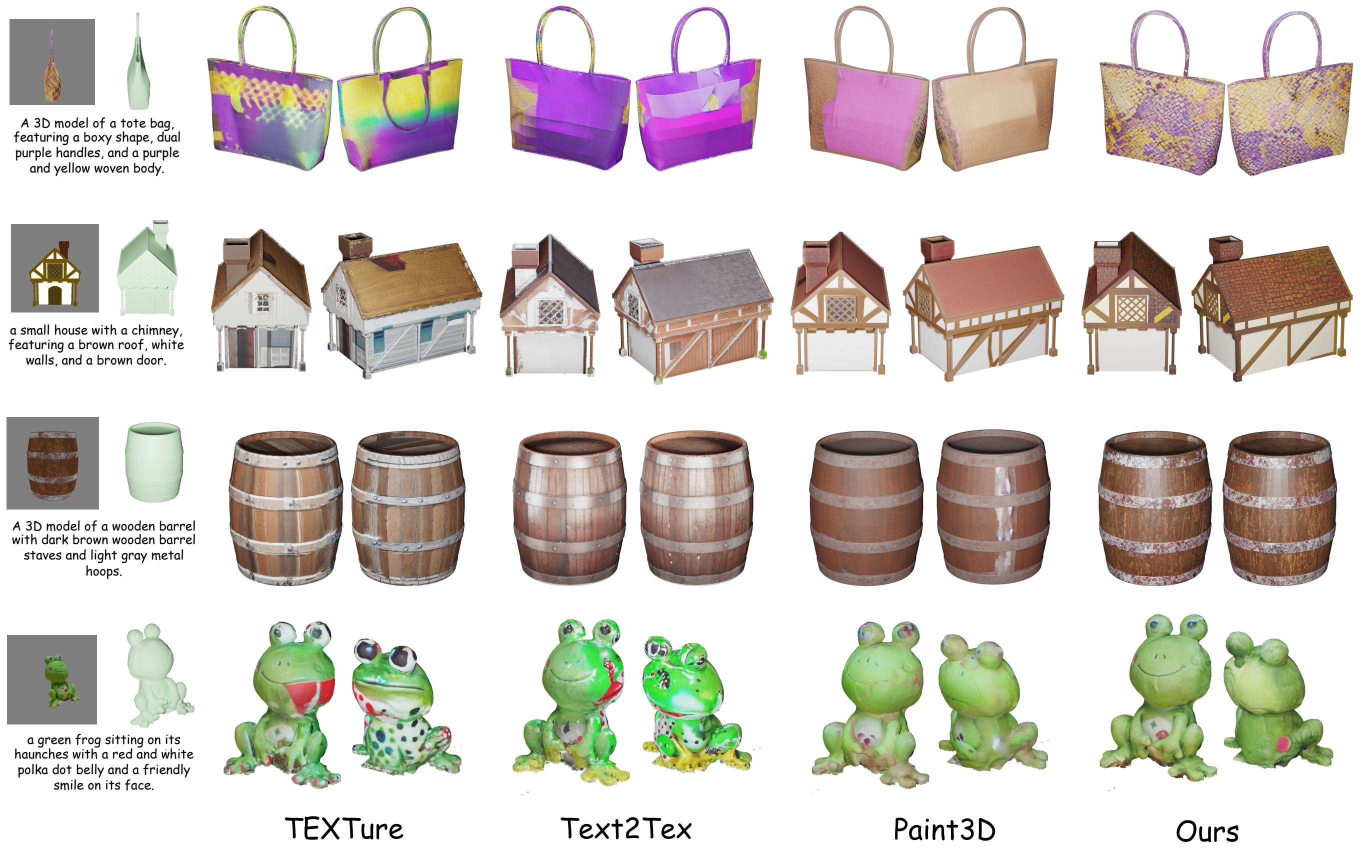

A 3D model of a tote bag, featuring a boxy shape, dual purple handles, and a purple and yellow woven body.
a small house with a chimney, featuring a brown roof, white walls, and a brown door.
A 3D model of a wooden barrel with dark brown wooden barrel staves and light gray metal hoops.
a green frog sitting on its haunches with a red and white polka dot belly and a friendly smile on its face.
TEXTure
Text2Tex
Paint3D
Ours

Fig. 6. **Comparison with state-of-the-art methods.** We compare our method with four representative state-of-the-art methods. Our model can synthesize more detailed and coherent textures compared to these methods which rely on test-time optimization using a 2D pretrained text-to-image diffusion model. Also, our method trained on the 3D dataset and 3D representation avoids the Janus problem that commonly occurs in other methods.

their quality. Besides, they do not well preserve the guided image information. In the fourth example, they face the Janus problem, where inappropriate features such as eyes and a mouth appear on both the front and back of the frog. In contrast, our method successfully generates textures with rich detail, maintains global coherence, and avoids the Janus problem.

Then, we compare our method with Paint3D, which utilizes a two-stage approach. In the first stage, similar to TEXTure and Text2Tex, Paint3D generates a texture map x^{coarse} through iterative inpainting; additionally, it trains a refinement model D to address unrealistic lighting issues and fill in areas that were not painted in the first stage. However, this model cannot be used standalone for texture generation and we observe an inevitable loss of texture details after the refinement stage. The results in fig. 6 show Paint3D produces over-smooth results and still exhibits the Janus problem.

Quantitative comparisons. We conduct quantitative comparisons on 400 test objects. Following [Siddiqui et al. 2022; Yu et al. 2023a], we render images from textured meshes and calculate the FID and KID with ground-truth images. As shown in table 1, our method significantly outperforms other methods. Additionally, we tested our model's runtime speed on a single A100 GPU, and our method is notably faster than others, completing evaluations in under 10 seconds without requiring test-time optimization.

Table 1. **Quantitative Comparisons.** FID and KID ($\times 10^{-4}$) are evaluated on multi-view renderings. Our method achieves state-of-the-art texture quality with significantly faster inference.

Methods	FID(↓)	KID(↓)	Time(↓)
TEXTure [Richardson et al. 2023]	48.31	48.00	80s
Text2Tex [Chen et al. 2023b]	49.85	47.38	344s
Paint3D [Zeng 2023]	43.55	25.73	95s
Ours	**34.53**	**11.94**	**10s**

It is worth noting that our method fundamentally differs from others as it is a feed-forward model. Consequently, we can further accelerate our model using techniques such as model compression or diffusion acceleration methods like consistency distillation, which we leave as future work.

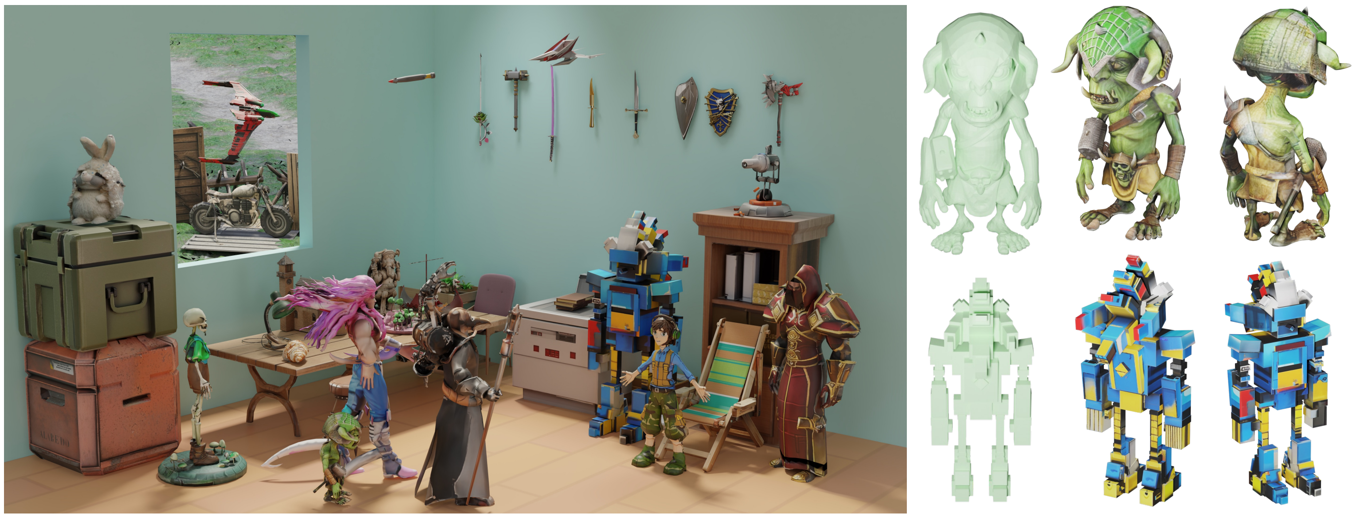


Fig. 7. **An indoor scene with all meshes textured by TEXGen.** We generate a single view using text-conditioned ControlNet with depth control for each mesh and paint them with both the text and single view prompt with TEXGen.

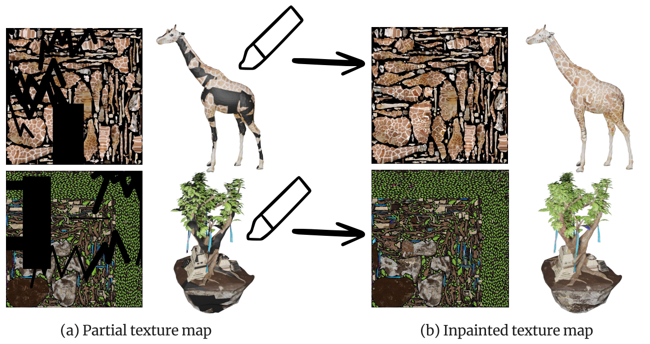


Fig. 8. **TEXGen as a texture inpainter.** We demonstrate the potential of TEXGen as a texture inpainter. We showcase here (a) randomly masked texture maps and (b) the inpainted texture maps, with unknown regions rendered as black.

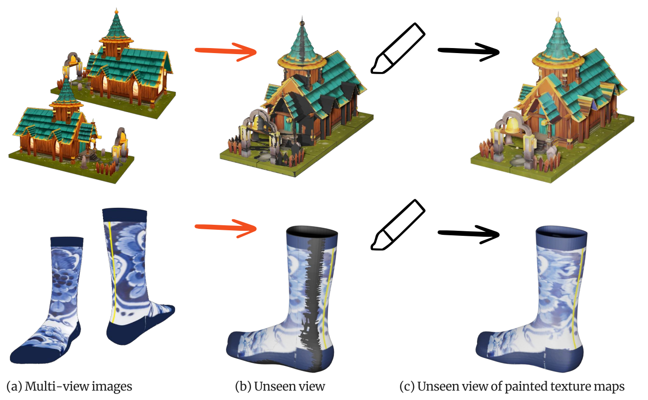


Fig. 9. **Texture completion from sparse views.** With sparse views of objects provided (front and back views as shown in (a)), unprojected textures retain many unseen regions (b). TEXGen effectively fills these unseen regions with harmonious textures (c).

5.2 Applications

Without any fine-tuning, our model serves as a powerful foundation for various applications. Firstly, we showcase the ability of our model to generate mesh textures conditioned solely on text prompts by integrating it with Depth-ControlNet [Zhang et al. 2023]. As shown in fig. 7, we compose a scene with the generated objects to showcase our results. The scene is vivid and highlights the potential of our model in scene texturing applications. Each object can be customized using individual text prompts for control.

We also evaluate the text-condition results by two ways. Firstly, we conducted a user study focused on text alignment and texture quality. Participants were asked in each round to choose the best of four texture results based on how well they matched the text description and the quality of the texture. A total of 423 responses were collected, and the final analysis, presented in Table 2, shows which algorithm produced the most preferred results. Besides, we employ the Multi-modal Large Language Model (MLLM) Score [Huang et al. 2023b] as an objective metric which provides a robust measure of the alignment between the input text prompt and the generated texture, especially under complex conditions that are close to real-world scenarios. As shown in Table 2, both human preference and MLLM score prove that our method outperforms other methods.

Then, we demonstrate that our model can flexibly paint different sources of partial texture maps. In fig. 8, we randomly mask sections of a texture map and use our model to fill in the gaps. The results show seamless integration with the existing texture. Additionally, in fig. 9, we consider a scenario where the user provides multi-view images. In such cases, there are often many unseen areas. Our model effectively paints these regions, ensuring continuity and coherence.

Table 2. **Quantitative evaluation on text-condition generation. *Preference*** refers to the comprehensive user study evaluating the alignment with the text description and the quality of the texture. For ***MLLM Score***, Claude 3.5-sonnet [Anthropic 2024], a state-of-the-art MLLM, is used to calculate the text-to-texture similarity scores. To be consistent with the conclusions in [Huang et al. 2023a], we use the same Chain-of-Thought prompts described in the study.

Methods	Paint3D	TEXTure	Text2Tex	Ours
Preference(%)(↑)	16.5	7.1	7.1	**69.3**
MLLM Score(↑)	64.8	69.8	64.8	**74.2**

5.3 Model Analysis

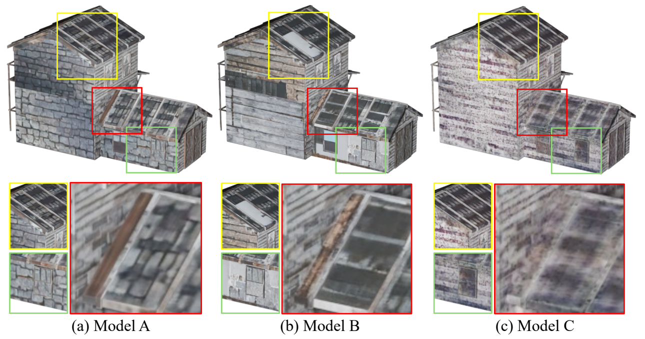


Fig. 10. **Qualitative ablation results on the hybrid design.** Compared to the full model A (a), the model B (b) with only UV blocks can not easily capture overall semantic and 3D consistency while that the model C (c) with only point blocks struggles with producing high-frequency patterns.

Hybrid blocks. We ablate the hybrid design of our block. For efficiency reasons, we select objects from the house category in our dataset for training and testing in this experiment. We use 10,000 models for training and 100 for testing. We reduce the model size by half to create Model A, which uses the hybrid block. Keeping the parameter count the same as Model A, we remove the point block and replace it with additional UV blocks to create Model B. Similarly, we remove the UV block and replace it with additional point blocks to create Model C, maintaining the same parameter count as Model A. We train all three models on 16 A100 GPUs, with a batch size of 64, for 85K iterations. As seen in fig. 10, compared to Model A, Model B generates textures with inconsistent styles, such as the white patches on the roof in the yellow zoom-in region and the seam artifacts on the wall in the green box. Model C produces relatively consistent textures, but the texture in the red zoom-in region lacks high-frequency details and appears blurred. We also tested FID and KID for the three models. As shown in the table 3, Model A performs the best, confirming the effectiveness of our hybrid block design.

Classifier-free guidance. A key advantage of our model is its use of diffusion-based training instead of a regression loss. This allows us to implement classifier-free guidance [Ho and Salimans 2022] during inference, which is crucial for enhancing texture synthesis quality. We found that the scale of the guidance weight significantly affects the results. While image diffusion models typically use a guidance weight of $\omega = 7.5$ to balance generation quality and condition alignment, our experiments show that a guidance weight of $\omega = 2.0$ provides the optimal balance for our model. As demonstrated in table 4, we varied the guidance weights and evaluated their performance in terms of FID and KID.

Table 3. **Quantitative ablation results on the hybrid design.** Starting from the full model, we build a UV block-only model (B) and a point block-only model (C) by replacing redundant blocks with additional ones, while maintaining the same number of model parameters. FID and KID ($\times 10^{-4}$) are evaluated on multi-view renderings.

Models/Metrics	FID(↓)	KID(↓)
Hybrid block (A)	**69.74**	**17.89**
w/o point block (B)	72.58	25.52
w/o UV block (C)	94.22	159.94

Table 4. **Ablation results of guidance weights.** We use different CFG weights to evaluate TEXGen, and the results show that the weight around 2-3 is optimal.FID and KID ($\times 10^{-4}$) are evaluated on multi-view renderings.

Metrics/ω	1	1.5	2	3	4	5	7.5
FID(↓)	35.01	34.73	**34.53**	35.19	35.69	36.69	39.58
KID(↓)	15.06	13.00	11.94	**11.71**	13.03	14.53	24.45

6 CONCLUSION

In this work, we have presented TEXGen, a large generative diffusion model designed for creating high-resolution textures for general 3D objects. TEXGen departs from conventional methods that depend on pre-trained 2D diffusion models that necessitate test-time optimization. Instead, our model efficiently synthesizes detailed and coherent textures directly, leveraging a novel hybrid 2D-3D block that adeptly manages both local detail fidelity and global 3D-aware interactions. Capable of generating high-resolution texture maps in a feed-forward manner, TEXGen supports a variety of zero-shot applications, including text-guided texture inpainting, sparse-view texture completion, and text-to-texture synthesis. As the first feed-forward model capable of generating textures for general objects, TEXGen sets a new benchmark in the field. We anticipate that our contributions will inspire and catalyze further research and advancements in texture generation and beyond.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).
Anthropic. 2024. Claude 3.5 Sonnet. https://www.anthropic.com/news/claude-3-5-sonnet Accessed: 2024-08-20.
James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. 2023. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf* 2, 3 (2023), 8.
Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. 2023. Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets. *CoRR* abs/2311.15127 (2023).
Alexey Bokhovkin, Shubham Tulsiani, and Angela Dai. 2023. Mesh2tex: Generating mesh textures from image queries. In *Proceedings of the IEEE/CVF International*

Conference on Computer Vision. 8918–8928.

Tianshi Cao, Karsten Kreis, Sanja Fidler, Nicholas Sharp, and Kangxue Yin. 2023. TexFusion: Synthesizing 3D Textures with Text-Guided Image Diffusion Models. In *ICCV*. IEEE, 4146–4158.

Duygu Ceylan, Valentin Deschaintre, Thibault Groueix, Rosalie Martin, Chun-Hao Huang, Romain Rouffet, Vladimir Kim, and Gaëtan Lassagne. 2024. MatAtlas: Text-driven Consistent Geometry Texturing and Material Assignment. *arXiv preprint arXiv:2404.02899* (2024).

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. 2015. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012* (2015).

Dave Zhenyu Chen, Yawar Siddiqui, Hsin-Ying Lee, Sergey Tulyakov, and Matthias Nießner. 2023b. Text2Tex: Text-driven Texture Synthesis via Diffusion Models. In *ICCV*. IEEE, 18512–18522.

Kevin Chen, Christopher B. Choy, Manolis Savva, Angel X. Chang, Thomas A. Funkhouser, and Silvio Savarese. 2018. Text2Shape: Generating Shapes from Natural Language by Learning Joint Embeddings. In *ACCV (3) (Lecture Notes in Computer Science, Vol. 11363)*. Springer, 100–116.

Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. 2023a. Fantasia3D: Disentangling Geometry and Appearance for High-quality Text-to-3D Content Creation. In *ICCV*. IEEE, 22189–22199.

An-Chieh Cheng, Xueting Li, Sifei Liu, and Xiaolong Wang. 2023. TUVF: Learning Generalizable Texture UV Radiance Fields. *CoRR* abs/2305.03040 (2023).

Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, and Chunhua Shen. 2021. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882* (2021).

Katherine Crowson, Stefan Andreas Baumann, Alex Birch, Tanishq Mathew Abraham, Daniel Z Kaplan, and Enrico Shippole. 2024. Scalable High-Resolution Pixel-Space Image Synthesis with Hourglass Diffusion Transformers. *arXiv preprint arXiv:2401.11605* (2024).

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. 2023. Objaverse: A Universe of Annotated 3D Objects. In *CVPR*. IEEE, 13142–13153.

Chenjian Gao, Boyan Jiang, Xinghui Li, Yingpeng Zhang, and Qian Yu. 2024. GenesisTex: Adapting Image Denoising Diffusion to Texture Space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 4620–4629.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2020. Generative adversarial networks. *Commun. ACM* 63, 11 (2020), 139–144.

David Hilbert and David Hilbert. 1935. *Neubegründung der mathematik. erste mitteilung.* Springer.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems* 33 (2020), 6840–6851.

Jonathan Ho and Tim Salimans. 2022. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598* (2022).

Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. 2023. Lrm: Large reconstruction model for single image to 3d. *arXiv preprint arXiv:2311.04400* (2023).

Kaiyi Huang, Chengqi Duan, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. 2023a. T2I-CompBench: A Comprehensive Benchmark for Text-to-Image Models. https://github.com/Karine-Huang/T2I-CompBench GitHub repository.

Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. 2023b. T2i-compbench: A comprehensive benchmark for open-world compositional text-to-image generation. *Advances in Neural Information Processing Systems* 36 (2023), 78723–78747.

Tero Karras, Samuli Laine, and Timo Aila. 2019. A Style-Based Generator Architecture for Generative Adversarial Networks. In *CVPR*. Computer Vision Foundation / IEEE, 4401–4410.

Xin Lai, Jianhui Liu, Li Jiang, Liwei Wang, Hengshuang Zhao, Shu Liu, Xiaojuan Qi, and Jiaya Jia. 2022. Stratified transformer for 3d point cloud segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 8500–8509.

Jiahao Li, Hao Tan, Kai Zhang, Zexiang Xu, Fujun Luan, Yinghao Xu, Yicong Hong, Kalyan Sunkavalli, Greg Shakhnarovich, and Sai Bi. 2023. Instant3d: Fast text-to-3d with sparse-view generation and large reconstruction model. *arXiv preprint arXiv:2311.06214* (2023).

Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. 2023. Magic3d: High-resolution text-to-3d content creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 300–309.

Shanchuan Lin, Bingchen Liu, Jiashi Li, and Xiao Yang. 2024. Common diffusion noise schedules and sample steps are flawed. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision.* 5404–5411.

Yuxin Liu, Minshan Xie, Hanyuan Liu, and Tien-Tsin Wong. 2023. Text-guided texturing by synchronized multi-view diffusion. *arXiv preprint arXiv:2311.12891* (2023).

Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. 2023. Latent-NeRF for Shape-Guided Generation of 3D Shapes and Textures. In *CVPR*. IEEE, 12663–12673.

Guy M Morton. 1966. A computer oriented geodetic data base and a new technique in file sequencing. (1966).

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

Michael Oechsle, Lars Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger. 2019. Texture fields: Learning texture representations in function space. In *Proceedings of the IEEE/CVF International Conference on Computer Vision.* 4531–4540.

William Peebles and Saining Xie. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision.* 4195–4205.

Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. 2022. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988* (2022).

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

Elad Richardson, Gal Metzer, Yuval Alaluf, Raja Giryes, and Daniel Cohen-Or. 2023. Texture: Text-guided texturing of 3d shapes. In *ACM SIGGRAPH 2023 Conference Proceedings.* 1–11.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* 10684–10695.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer, 234–241.

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. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems* 35 (2022), 36479–36494.

Tim Salimans and Jonathan Ho. 2022. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512* (2022).

Yawar Siddiqui, Justus Thies, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. 2022. Texturify: Generating Textures on 3D Shape Surfaces. *arXiv preprint arXiv:2204.02411* (2022).

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502* (2020).

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805* (2023).

Dmitry Tochilkin, David Pankratz, Zexiang Liu, Zixuan Huang, Adam Letts, Yangguang Li, Ding Liang, Christian Laforte, Varun Jampani, and Yan-Pei Cao. 2024. Triposr: Fast 3d object reconstruction from a single image. *arXiv preprint arXiv:2403.02151* (2024).

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* (2023).

Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A. Yeh, and Greg Shakhnarovich. 2023a. Score Jacobian Chaining: Lifting Pretrained 2D Diffusion Models for 3D Generation. In *CVPR*. IEEE, 12619–12629.

Peng Wang, Hao Tan, Sai Bi, Yinghao Xu, Fujun Luan, Kalyan Sunkavalli, Wenping Wang, Zexiang Xu, and Kai Zhang. 2023c. Pf-lrm: Pose-free large reconstruction model for joint pose and shape prediction. *arXiv preprint arXiv:2311.12024* (2023).

Peng-Shuai Wang. 2023. Octformer: Octree-based transformers for 3d point clouds. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–11.

Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. 2023b. ProlificDreamer: High-Fidelity and Diverse Text-to-3D Generation with Variational Score Distillation. *arXiv preprint arXiv:2305.16213* (2023).

Jinbo Wu, Xing Liu, Chenming Wu, Xiaobo Gao, Jialun Liu, Xinqi Liu, Chen Zhao, Haocheng Feng, Errui Ding, and Jingdong Wang. 2024. TexRO: Generating Delicate Textures of 3D Models by Recursive Optimization. *arXiv preprint arXiv:2403.15009* (2024).

Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. 2023. Point transformer v3: Simpler, faster, stronger. *arXiv preprint arXiv:2312.10035* (2023).

Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. 2022. Point transformer v2: Grouped vector attention and partition-based pooling. *Advances in Neural Information Processing Systems* 35 (2022), 33330–33342.

Yinghao Xu, Zifan Shi, Wang Yifan, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wetzstein. 2024. GRM: Large Gaussian Reconstruction Model for Efficient 3D Reconstruction and Generation. *CoRR* abs/2403.14621 (2024).

Yinghao Xu, Hao Tan, Fujun Luan, Sai Bi, Peng Wang, Jiahao Li, Zifan Shi, Kalyan Sunkavalli, Gordon Wetzstein, Zexiang Xu, et al. 2023. Dmv3d: Denoising multi-view diffusion using 3d large reconstruction model. *arXiv preprint arXiv:2311.09217*

(2023).

Yu-Qi Yang, Yu-Xiao Guo, Jian-Yu Xiong, Yang Liu, Hao Pan, Peng-Shuai Wang, Xin Tong, and Baining Guo. 2023. Swin3d: A pretrained transformer backbone for 3d indoor scene understanding. *arXiv preprint arXiv:2304.06906* (2023).

Yu-Ying Yeh, Jia-Bin Huang, Changil Kim, Lei Xiao, Thu Nguyen-Phuoc, Numair Khan, Cheng Zhang, Manmohan Chandraker, Carl S Marshall, Zhao Dong, et al. 2024. Texturedreamer: Image-guided texture synthesis through geometry-aware diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4304–4314.

Jonathan Young. 2018. xatlas. https://github.com/jpcy/xatlas.

Xin Yu, Peng Dai, Wenbo Li, Lan Ma, Zhengzhe Liu, and Xiaojuan Qi. 2023a. Texture Generation on 3D Meshes with Point-UV Diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 4206–4216.

Xin Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Song-Hai Zhang, and Xiaojuan Qi. 2023b. Text-to-3d with classifier score distillation. *arXiv preprint arXiv:2310.19415* (2023).

Xianfang Zeng. 2023. Paint3D: Paint Anything 3D with Lighting-Less Texture Diffusion Models. *arXiv preprint arXiv:2312.13913* (2023).

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 3836–3847.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 586–595.

Shangzhan Zhang, Sida Peng, Tao Xu, Yuanbo Yang, Tianrun Chen, Nan Xue, Yujun Shen, Hujun Bao, Ruizhen Hu, and Xiaowei Zhou. 2024. MaPa: Text-driven Photorealistic Material Painting for 3D Shapes. In *ACM SIGGRAPH 2024 Conference Papers*. 1–12.

Zi-Xin Zou, Zhipeng Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Yan-Pei Cao, and Song-Hai Zhang. 2023. Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers. *arXiv preprint arXiv:2312.09147* (2023).

A APPENDIX

A.1 Implementation Details

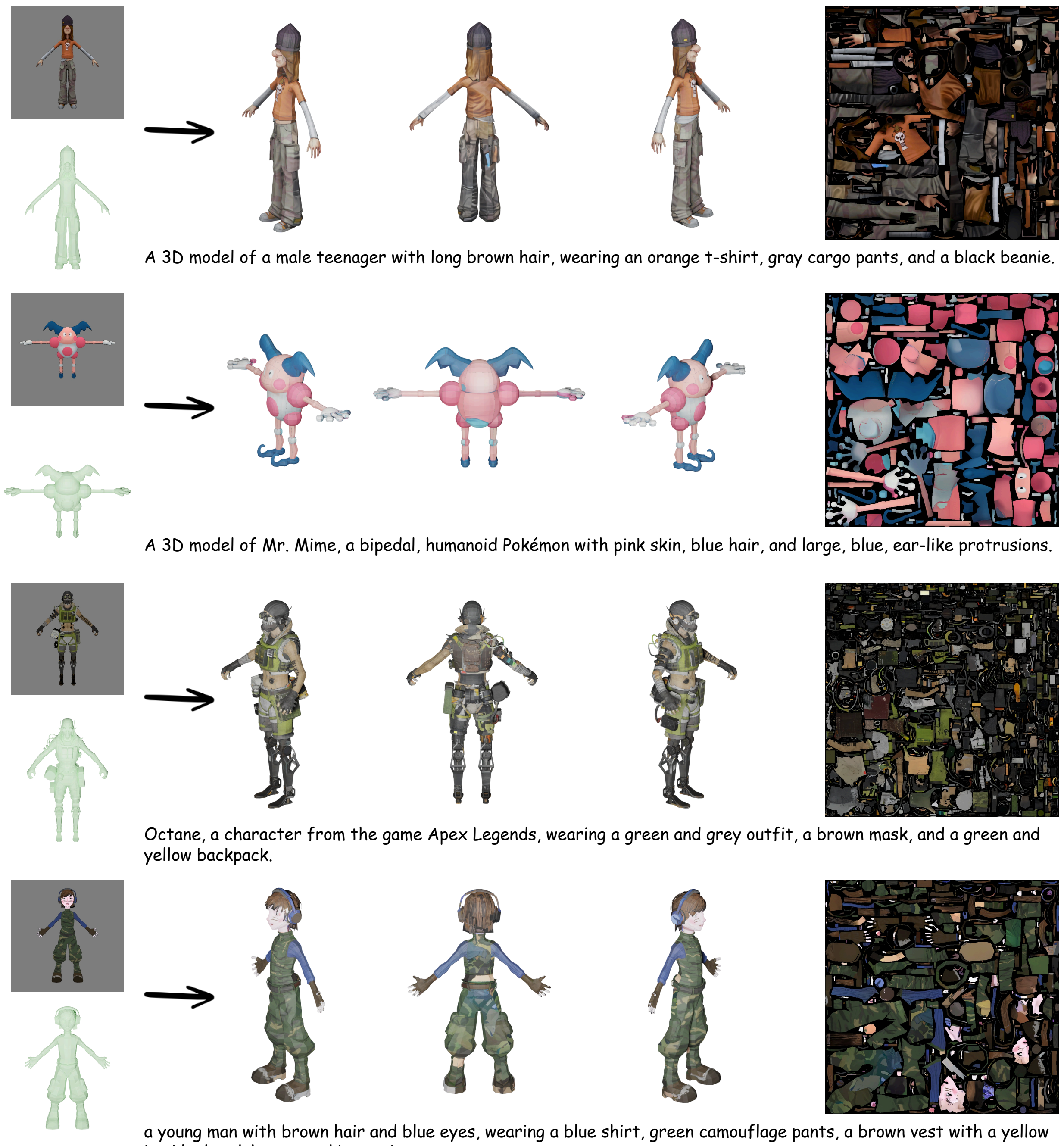
We use Objaverse [Deitke et al. 2023] as our raw data source, which contains over 800K 3D meshes. However, the texture structure of these meshes is not uniform and requires processing and filtering. For instance, some meshes are divided into parts with multiple texture images, while others have only base color information without texture images. To clean and reorganize the data, we first filter out meshes with poor texture quality. For the remaining meshes, we use xAtlas [Young 2018] to re-unfold the UVs, ensuring they are represented by a single UV atlas. Then, we bake the diffuse color from the original mesh files onto the newly parameterized meshes. Furthermore, we use Gemini [Team et al. 2023] to acquire each mesh's caption based on their rendered images. Ultimately, we processed and obtained 120,400 meshes with their corresponding texture images, using 120,000 for training and 400 for evaluation. We use five stages to construct our network (i.e., four downsampling and four upsampling stages). For efficiency, we use only UV blocks in the first stage. In the second stage, we use hybrid blocks but replace point attention with sparse convolution. In the remaining three stages, we use our designed hybrid blocks, with the attention layers having 2, 4, and 6 layers, respectively. The number of channels for each stage is 32, 256, 1024, 1024, and 2048, respectively. We use grid sizes of 0.02 and 0.05 in the sparse convolution blocks in the second stage and the attention blocks of the last three stages. The serialized point patch sizes are set to 256, 512, and 1024 in the final three stages. We train our model using the AdamW optimizer with $\beta_1 = 0.9, \beta_2 = 0.999$, and a weight decay of 0.05. The training process takes place on 32 A100 GPUs, with a total batch size of 64, spanning 400K iterations. A cosine scheduler is used to reduce the learning rate from 2e-4 to zero. The code is available at https://github.com/anonymous4coderelease/TEXGen.

A.2 More Qualitative Results

Firstly, in the video file submitted as supplementary material, we provide a video of the mesh rendering to demonstrate the results of our model. Additionally, our method demonstrates the ability to avoid the Janus problem and is applicable to real-scan models. The Janus problem in texture generation refers to the unintended duplication of features such as eyes and noses on both sides of 3D assets, particularly human faces. This issue typically occurs when methods lack 3D-awareness, often relying on pre-trained image generation models to independently generate textures for different views. Since our model is trained on 3D data and directly produces the full UV map in a forward manner, it effectively prevents this issue, as illustrated in fig. 11. Real-scan models often present unique challenges due to their non-smooth surfaces and fragmented, irregular UV maps. Despite these complexities, our method proves robust in handling real-scan models, effectively managing to generate high-quality texture maps that maintain the details and authenticity of the original objects, as demonstrated in fig. 12.

A.3 Limitations and Discussions

Currently, the condition images used during the training of our model are pose-aligned and shape-aligned, which may not meet the needs of users who wish to "transfer" textures using arbitrary images. However, we believe our network architecture could potentially handle such scenarios by incorporating dense image information through mechanisms like cross-attention, rather than relying on pixel projection. The primary challenge remains in constructing a suitable dataset for this purpose. As a future work, we plan to extend our model to generate Physically Based Rendering (PBR) material maps. It can be achieved by collecting and processing data necessary to train the model to generate PBR maps.



A 3D model of a male teenager with long brown hair, wearing an orange t-shirt, gray cargo pants, and a black beanie.

A 3D model of Mr. Mime, a bipedal, humanoid Pokémon with pink skin, blue hair, and large, blue, ear-like protrusions.

Octane, a character from the game Apex Legends, wearing a green and grey outfit, a brown mask, and a green and yellow backpack.

a young man with brown hair and blue eyes, wearing a blue shirt, green camouflage pants, a brown vest with a yellow buckle, headphones, and brown boots.

Fig. 11. **Results on 3D avatars.** Our model, trained on 3D data, adeptly avoids the Janus problem.

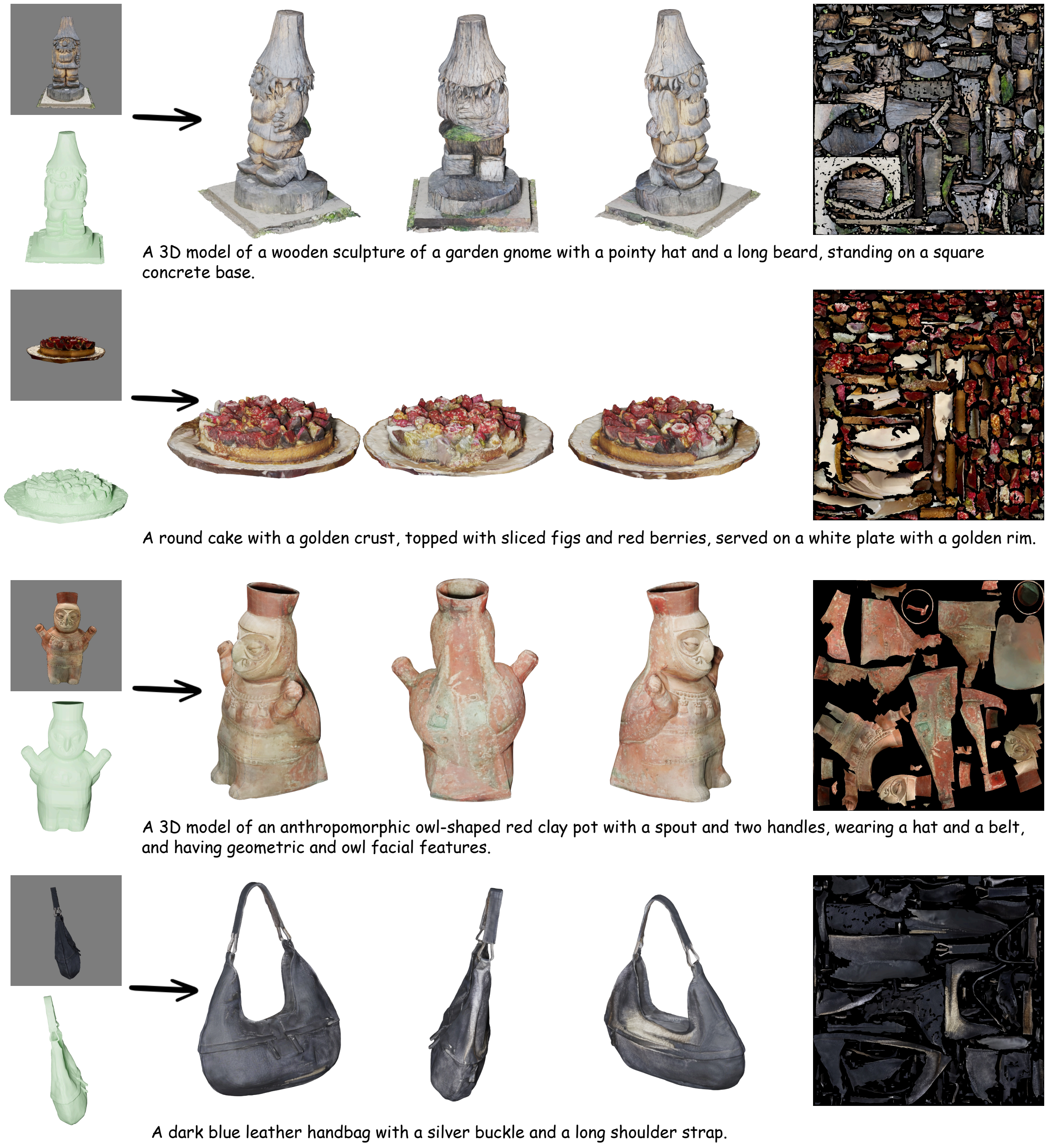

A 3D model of a wooden sculpture of a garden gnome with a pointy hat and a long beard, standing on a square concrete base.
A round cake with a golden crust, topped with sliced figs and red berries, served on a white plate with a golden rim.
A 3D model of an anthropomorphic owl-shaped red clay pot with a spout and two handles, wearing a hat and a belt, and having geometric and owl facial features.
A dark blue leather handbag with a silver buckle and a long shoulder strap.

Fig. 12. **Results on real-scan models.** Our method is robust to real-scan models with non-smooth surfaces and fragmented UV maps.