This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

UniTEX: Universal High Fidelity Generative Texturing for 3D Shapes

Yixun Liang[1,2*] Kunming Luo[1,2*] Xiao Chen[2*]
Rui Chen[1] Hongyu Yan[1] Weiyu Li[1,2] Jiarui Liu[1,2] Fei-Peng Tian[2] Ping Tan[1,2†]
[1]The Hong Kong University of Science and Technology [2]Light Illusions

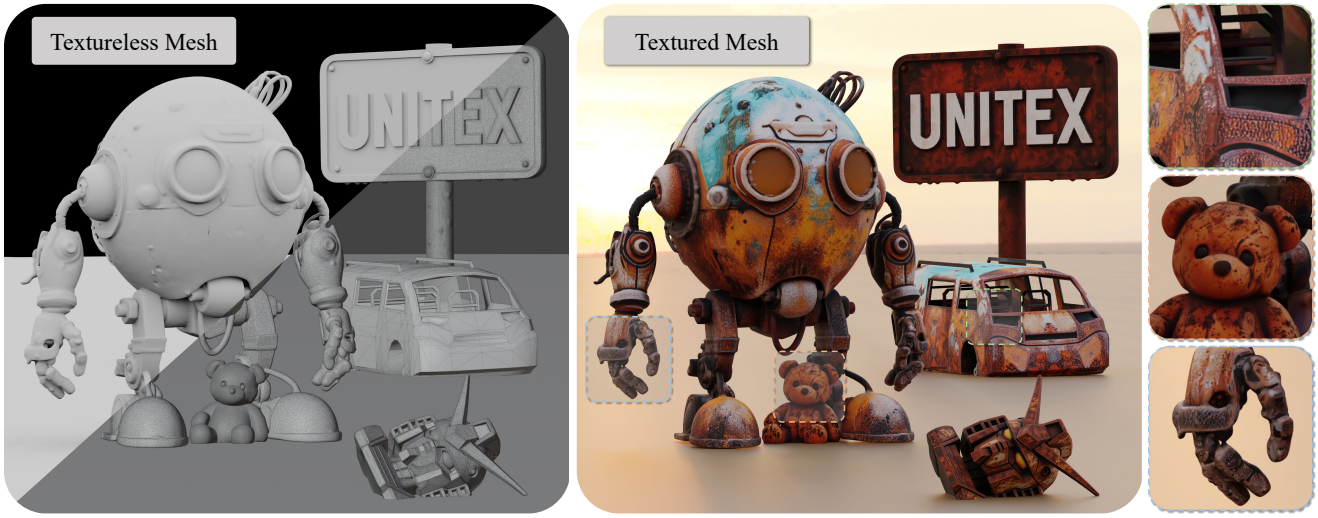


Figure 1. UniTEX generates *high-quality* and *complete* textures for both artist-created low-polygon mesh (the van shell) and generative high-polygon meshes (robots, toy bear, bust, and roadsign).

Abstract

We present UniTEX, a novel two-stage 3D texture generation framework to create high-quality, consistent textures for 3D assets. Existing approaches predominantly rely on UV-based models in the second stage to refine textures after reprojecting the generated multi-view images onto the 3D shapes, which introduces challenges related to topological ambiguity. To address this, we bypass the limitations of UV mapping by introducing a Large Texturing Model (LTM) that directly regresses textures in a unified 3D functional space. Moreover, to enable more effective and complete supervision of LTM, we propose to extend surface-defined textures into a continuous volumetric field to serve as an advanced training objective, which we refer to as Texture Functions (TF). Finally, we develop an advanced LoRA-based strategy for efficiently adapting large-scale 2D Diffusion Transformers (DiTs) for high-quality multi-view texture synthesis as our first stage. Extensive experiments demonstrate that UniTEX achieves superior visual quality and texture integrity compared to existing approaches, offering a generalizable and scalable solution for automated 3D texture generation. Code is available in: https://github.com/HKUST-SAIL/UniTEX

1. Introduction

High-quality texture generation plays a critical role in 3D asset creation, as it directly impacts both the visual realism and semantic fidelity of rendered models. In modern applications such as gaming, virtual reality, and digital content creation, photorealistic textures are essential for delivering immersive and visually engaging experiences. However, producing detailed and aesthetically refined textures typically requires significant manual effort and domain expertise, making the process time-consuming and resource-intensive.

Recently, diffusion models trained on large-scale datasets [11, 34] have brought about a revolutionary transformation in the field of image and video generation. Inspired by this, there is an expectation that 3D texture generation can also be greatly simplified, enabling direct generation from text descriptions or a single reference image. Prior works [1, 8, 27, 44] have primarily addressed this by adapting 2D generative models for multi-view texture generation and then projecting these multi-view textures onto the 3D mesh. While this approach leverages strong 2D priors and

*Equal contribution.
†Corresponding author.

achieves some promising results, the generated textures often suffer from issues of self-occlusion and multi-view inconsistencies, leading to textures on the 3D surface that are incomplete or fragmented. To mitigate this, some early attempts [1, 43, 44] introduce a second stage of UV-based inpainting process to merge the partial views into a coherent texture map to handle those incomplete or inconsistent areas after multi-view projection. Although these methods provide completed texture, in practice, UV-based inpainting often struggles to handle meshes created from generative AI pipelines [22–24, 29, 41, 51] as shown in Fig. 2. This practical observation highlights the generalization bottlenecks of UV-based inpainting for real-world 3D texture generation.

This challenge stems from a fundamental limitation of UV mapping, which we refer to as **topological ambiguity**. Specifically, a single 3D mesh can correspond to multiple valid UV layouts that are not uniquely determined by the geometry but are also highly sensitive to vertices/faces distributions and UV unwarping algorithms. This ambiguity poses a critical challenge for UV-based inpainting models because there is no guarantee that the UV parameterization used during inference will match that of the training stage. Therefore, the quality of the results is not guaranteed. (We provide an additional experiment to prove that, which about UV-based methods under different uv-parameterizations in Supp. B.1.) This problem becomes especially pronounced when texturing meshes produced by generative pipelines [22, 23, 51], which are typically extracted using MarchingCubes [28] and exhibit diverse mesh topologies. The resulting domain gap severely limits generalization during the inference of current UV-based inpainting models.

To address this challenge, we formulate texture inpainting and prediction as a native 3D regression task that takes both images and geometry as input. Accordingly, we propose the *Large Texturing Model (LTM)* (detailed in Sec. 3.3), which learns textures within a unified 3D functional space from both images and geometry. Moreover, we introduce the *Texture Function (TF)*, which extrapolates texture signals, originally defined only on object surfaces, into the whole 3D space, similar to signed or unsigned distance fields (SDFs/UDFs) [2]. This transforms the surface texture signal into a smooth and continuous volumetric field, analogous to how SDFs/UDFs represent geometry. The resulting TF serves as a training objective for LTM, providing richer and more complete supervision. By removing dependence on UV layouts and introducing a more advanced supervision signal, our method reduces the domain gap in texture generation between training and real-world data, offering a generalizable and scalable second stage in the 3D texturing pipeline.

To further enhance the overall quality of the texture generation pipeline, we propose an advanced LoRA-based

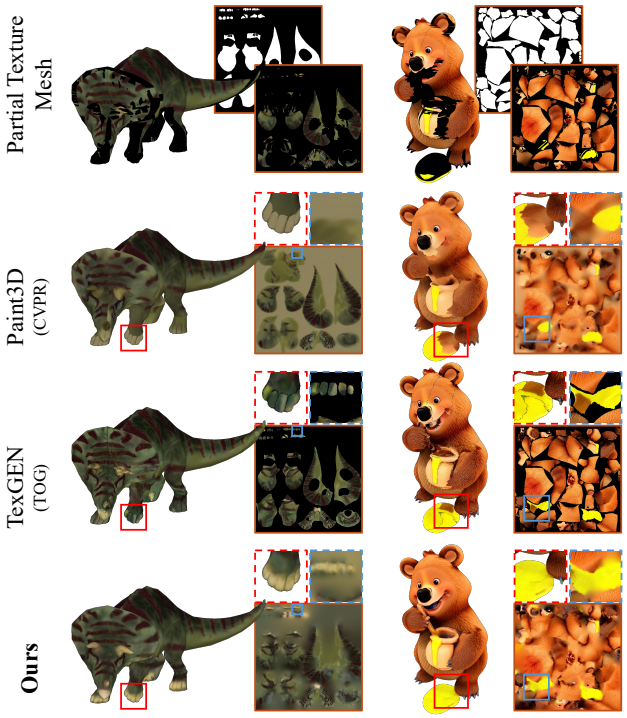


Figure 2. UV-based texturing models perform well on in-domain, artist-created meshes (first column), but struggle with out-of-distribution, generated meshes (second column). While effective on large, continuous regions, they fail to handle small, fragmented areas due to training biases toward clean, large-region UV layouts. In contrast, our method operates outside the UV space, enabling better generalization across diverse mesh types. Additional comparisons are provided in Sec. 4.3.

training strategy that efficiently adapts large-scale diffusion transformers (DiTs) for multi-view synthesis conditioned on geometry and reference images. Since first-stage texture generation heavily depends on the capabilities of 2D diffusion models—and large-scale DiTs currently dominate 2D generative modeling—our approach is specifically designed to leverage these strengths effectively.

We evaluate our method through extensive experiments on multiple settings, demonstrating its superiority over existing approaches in terms of both visual quality and texture integrity. Overall, our contributions are summarized as:

- We design a novel *Large Texturing Model (LTM)* to directly predict texture from images and geometry inputs in 3D space, which bypasses UV mapping and models texture as a completed spatial field.
- We propose *Texture Function (TF)*, an advanced continuous 3D texture representation that serves as a more effective supervision signal for LTM.
- We develop a LoRA-based strategy to efficiently adapt large diffusion transformers (DiTs) for downstream tasks, enabling high-quality multi-view synthesis for texture generation using large-scale Diffusion Transformers.

2. Related work

2.1. 3D Texturing using Diffusion Models

Most existing texturing methods leverage 2D priors from text-to-image (T2I) diffusion models [34] to generate textures from text or images. Methods like Text2Tex [4] and TEXture [33] iteratively paint meshes from multiple views but often lack multi-view consistency. Another line of work uses optimization-based approaches with Score Distillation Sampling (SDS) to directly refine texture maps, as seen in Fantasia3D [5], Magic3D [25], and DreamMat [49]. FlashTex [9] improves light disentanglement by introducing a light-conditioned diffusion model in a two-stage pipeline. However, SDS-based methods typically rely on single-view diffusion models, suffer from view inconsistency (e.g., Janus problem), and require expensive iterative optimization, limiting their scalability.

Subsequent works aim to improve multi-view consistency and texture quality by fine-tuning diffusion models [8, 44, 47, 51], introducing advanced sampling strategies [3, 27, 46], or designing 2D diffusion models to adopt UV refinement [1, 43, 44]. TexFusion [3] uses a denoising sampler that fuses images across views in UV space at each step. Paint3D [44] refines textures through a UV inpainting model. Meta 3D TextureGen [1] leverages a geometry-aware T2I model for consistent multi-view synthesis, and Hunyuan3D 2.0 [51] proposes a model to generate multi-view textures simultaneously. Although these UV-based methods can leverage the powerful priors of 2D diffusion models, the UV representation inherently suffers from topological ambiguity, which limits the generalization capabilities of these methods. In this paper, we propose Texture Functions (TFs), a continuous 3D texture representation, aiming to bypass the limitations of UV unwrapping and achieve complete and high-quality texturing.

2.2. 3D Native Texturing

Another line of work trains generative models directly on 3D data with ground-truth textures [26, 30, 40, 42, 43]. Texture Field [30] learns implicit color fields on 3D surfaces, while Texturify [35] uses face convolutions and adversarial rendering losses for per-face texture prediction. Recent diffusion-based methods such as PointUV [42], TexGEN [43], and TexOct [26] generate point cloud colors mapped to UV textures. TexGaussian [40] introduces an octree-based 3D Gaussian representation trained to predict textures from text and geometry and then bake them to the original mesh. Although these native 3D approaches offer better completion compared with 2D diffusion-based methods, they are limited by the scarcity of 3D data and limited generalizability. In this paper, we reposition native 3D texturing as a refinement module in a second-stage refinement and get more generative conditions (multi-view images) from 2D foundation models like [20]. By integrating the strong generative capabilities of 2D diffusion models, our method achieves more robust performance under limited data and produces higher-quality results.

3. Methods

3.1. Overview

In this section, we present a detailed design of our texturing pipeline, which is a **two-stage** pipeline that fully considers the combination of the 2D diffusion models for multi-view generation and 3D texturing methods for texture completion. As shown in Fig. 3, given a single input image and a textureless 3D mesh, our system first fine-tunes two large-scale diffusion transformers (Flux [*]) using an efficient LoRA-based training strategy (Sec. 3.2) to generate six orthographic, illumination-free views conditioned on the mesh's normals and canonical coordinate map (CCM) [21, 39], which is a special type of rendered image where each pixel encodes the 3D coordinate of the surface point it corresponds to). These generated images can optionally be used for super-resolution [10]. After reprojection and blending, the synthesized views, along with the partially textured geometry, are fed into our Large Texturing Model (LTM) (Sec. 3.3), which predicts the corresponding complete texture functions(Sec. 3.3.2). The final texture is then synthesized by blending the predicted texture functions with the initial partial textured geometry.

3.2. Efficient DiT Tuning for MV Generation

In this section, we present the details of the first stage in Fig 3. We fine-tune two large-scale diffusion transformers using in-context learning to adapt the model to our specific task: 3D texturing. Specifically, the first Flux takes reference images, normal maps, and canonical coordinate maps (CCMs) of a textureless mesh as input and generates corresponding shaded images. The second Flux is used to delight and generate diffuse color for final texturing. The generated diffuse color images can be used with super-resolution [10] for better results.

However, effectively adapting 2D diffusion models for 3D texturing presents unique challenges. Unlike traditional 2D generation, our task requires significantly more conditioning signals. For instance, generating six-view images at 512×512 resolution demands a reference image and corresponding geometric information (e.g., normals and CCMs) for all six views. Moreover, since diffusion transformers (DiTs) rely heavily on in-context learning [36, 50]—where all inputs and conditions are jointly encoded as tokens—the resulting volume of token inputs substantially increases training cost and slows convergence.

[*]https://huggingface.co/black-forest-labs/FLUX.1-dev

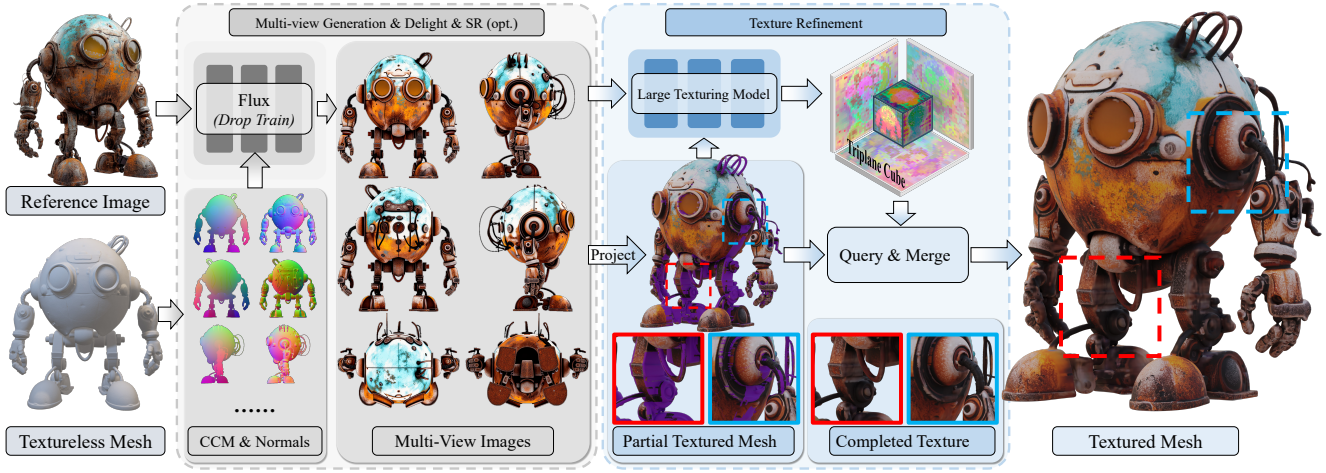


Figure 3. Overall pipeline of UniTEX. Given a textureless geometry and reference image, UniTEX first generates a high-fidelity multi-view image through three steps (RGB generation, Delighting, and super-resolution (SR)) using finetuned DiTs (detailed in Sec. 3.2). The texture will be reprojected to a partial textured mesh and sent to the Large Texturing Model (Detailed in Sec. 3.3) with generated images to predict the corresponding complete texture functions (detailed in Sec. 3.3.2). The final texture is then synthesized by blending the predicted texture functions with the partial textured geometry.

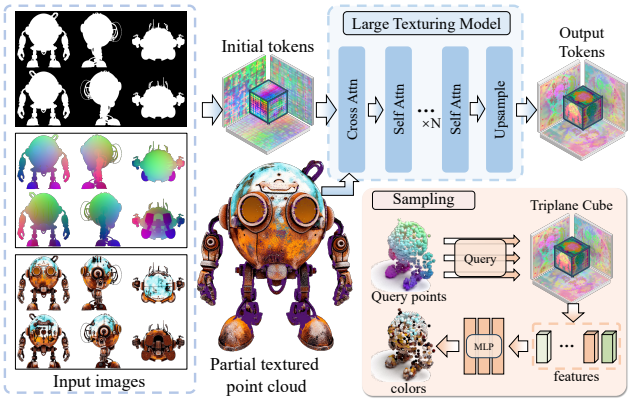


Figure 4. Pipeline of the Large Texturing Model. Given a partially textured geometry and six input views, we first unify them into a shared triplane-cube token representation. A transformer-based architecture then processes these tokens to extract geometry-aware features, which are subsequently decoded into colors using a lightweight MLP.

Recent studies, such as MVDiffusion++[37] and Long-LoRA[7], suggest that it is not strictly necessary to preserve the full forward computation pattern during both training and inference when fine-tuning large-scale foundation models. Specifically, Long-LoRA proposes truncating attention windows during training while restoring full attention at inference time to save resources and achieve long context finetuning. Similarly, MVDiffusion++ demonstrates that training on fewer views and inferring more views at test time can still yield strong performance. This invites a rethinking of why such behaviors can work and what fine-tuning is truly optimizing for. Using texture generation as an example, the image quality does not need to be trained during finetuning, the pattern we want the model to learn is "multi-view consistency and correspond to conditions". Learning such a pattern may not require simultaneous access to all input tokens. Building on this insight, we propose a drop training strategy: during each training step, only a subset of all tokens is retained, and the diffusion transformer is conditioned and generated solely on these selected tokens rather than the full input tokens. This approach reduces the dependency on complete image tokens, allowing the model to learn from partial information while maintaining task-relevant needs. As shown in Sec. 4.5.1, This method achieves comparable generation quality to full-input fine-tuning at the same iterations while significantly accelerating training and reducing the computational cost.

3.3. Large Texturing Model

Previous two-stage texturing methods primarily relied on UV-based inpainting, which often suffers from topological ambiguities and leads to suboptimal texture quality. In our work, we propose *Large Texturing Model (LTM)* to regress the texture in a 3D functional space to bypass the topology ambiguity and serve as the second stage in Fig. 3.

The pipeline of our *Large Texturing Model (LTM)* is shown in Fig. 4. Based on the generated images from Sec. 3.2 with textureless / incomplete textured geometry, we introduce the *Large Texturing Model (LTM)* (detailed in Sec. 3.3.1), which is designed to regress the *Texture Functions* (Detailed in Sec. 3.3.2) for the second stage of our texturing pipeline.

3.3.1. Architecture Design

We first initialize our *triplane-cube* with a set of learnable tokens, which contain a high-resolution triplane $\mathcal{T} \in \mathbb{R}^{3\times32\times32}$ as well as a low-resolution cube $\mathcal{C} \in \mathbb{R}^{8\times8\times8}$,

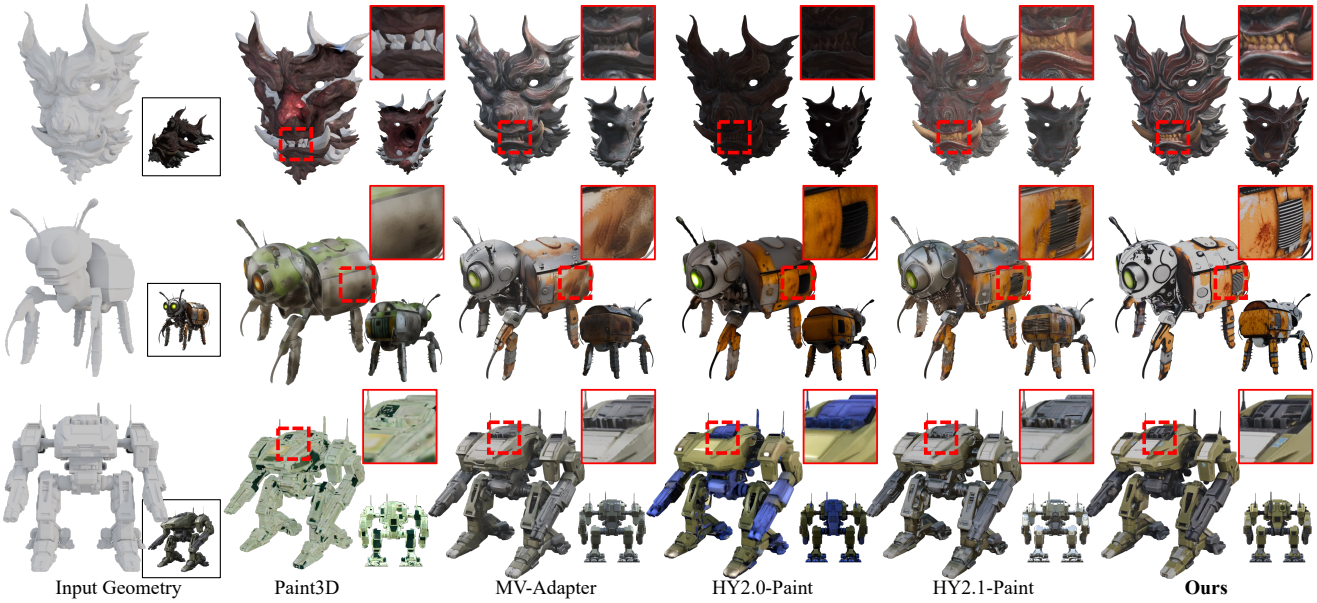


Figure 5. Qualitative comparison of our method against SoTA open-source texturing methods. Our method consistently achieves superior texture quality and generalization across diverse mesh sources. (Detailed analysis is provided in Sec. 4.2; best viewed by zooming in.)

which have been extensively demonstrated to offer a superior representation compared to triplanes [13, 32]. Then, we encode 6 ortho-view images, CCM, and alpha maps to a triplane like CRM [39] and add them together to the initial triplane tokens. After that, we encode geometry information into such a representation following Shape2VecSet [45]. Specifically, we sample the colored point cloud in the partial textured geometry and use the *triplane-cube* tokens to query them through cross-attention.

After integrating the information of 2D images and 3D geometry, we use a transformer architecture with self-attention to process these flattened features of *triplane-cube*. Then, the final output is first reshaped into triplane and cube representations, then upsampled using 2D and 3D deconvolution layers respectively.

During texture sampling, we implement an MLP, denoted as MLP_θ to predict RGB queried from the triplane-cube features denoted as $\mathcal{TC}$. Specifically, given a query 3D point $\boldsymbol{x}$, we predict the correspondence color as:

$$\hat{\mathbf{c}}(\boldsymbol{x}, \mathcal{TC}) = \text{MLP}_\theta(gridsample(\boldsymbol{x}, \mathcal{TC})), \quad (1)$$

where $\hat{\mathbf{c}} \in \mathbb{R}^3$ is the predicted RGB color for point $\boldsymbol{x}$. Therefore, after training, we can achieve texture prediction/inpainting through querying the color of corresponding spatial points.

3.3.2. Training Objective – Texture Functions:

In this section, we describe how we train our LTM. Unlike native 3D geometry generation or reconstruction, which benefits from well-defined Signed or Unsigned Distance Fields (SDF/UDF) in 3D functional space as continuous and complete supervision signals [22, 23, 51], texture is traditionally defined only on the surface of 3D objects. Prior works [14, 18, 20, 30, 35] typically rely on surface or volume rendering to supervise the texture representation via 2D projections. However, this form of supervision is inherently sparse and limited in coverage compared to the dense volumetric supervision available in geometry tasks. As shown in geometry generation, complete 3D supervision—such as that provided by SDFs/UDFs—consistently outperforms sparse alternatives like LRM-based methods.

Motivated by this insight, we propose to extend texture from a surface-restricted signal to a continuous volumetric function defined throughout the 3D space as an advanced supervision signal. Formally, similar to UDF extractions and proposed in [2], we define a Texture Function as a mapping over 3D coordinates $\boldsymbol{x} \in \mathbb{R}^3$, where each texture value (RGB) is obtained by finding the closest point on the mesh surface Ω to $\boldsymbol{x}$ and querying its corresponding color. This definition aligns naturally with volumetric geometry representations and enables unified learning across the entire 3D domain.

With such texture functions, the training objectives of LTM are defined as:

$$\mathcal{L}_{\text{texture}} = \mathbb{E}_{\mathbf{x}\sim\Omega}\left[\|\hat{\mathbf{c}}(\boldsymbol{x}, \mathcal{TC}) - \mathbf{c}(\boldsymbol{x})^*\|^2\right] + \lambda\mathcal{L}_{tv}(\mathcal{T}), \quad (2)$$

where $\boldsymbol{x}$ is the set of surface points, $\hat{\mathbf{c}}(\mathbf{x})$ is the predicted color, and $\mathbf{c}(\boldsymbol{x})$ is the ground truth color derived from texture functions. $\mathcal{L}_{tv}$ is denoted as the total-variance loss to regularize the variance of grid-based 3D representation. The λ denotes the weight, which we set as 0.0005 in our experiments. Notably, we use truncated texture functions like truncated SDF that are used for training in [6, 23, 51]. We set the threshold as 0.025. For those points that are out

Table 1. Quantitative comparison between artist-created and generative mesh on different texturing methods.

Methods.	Artist-created Mesh				Generative Mesh	
	CMMD↓	FID_{CLIP}↓	$CLIP_{score}$↑	LPIPS↓	$CLIP_{score}$↑	User-perf.↑
Paint3D	1.196	20.52	0.840	0.107	0.720	6.82%
TexPainter	1.632	27.13	0.804	0.112	0.709	1.36%
TexGaussians	1.290	21.08	0.836	0.095	0.681	4.55%
Hunyuan3D-Paint	0.909	20.38	0.824	0.091	0.800	21.36%
Ours	**0.826**	**16.03**	**0.844**	**0.090**	**0.807**	**65.91%**

of the truncated value, we use the background color as the ground truth for training.

In addition to providing surface point supervision as in previous methods, our approach further extends supervision to off-surface regions in 3D space. Specifically, the colored regions are expanded into a thin shell (after truncation) surrounding the geometry. This design offers several advantages. First, during color prediction, it implicitly encourages the model to develop a volumetric understanding of the 3D object. Second, the thin shell reduces reliance on highly precise mesh modeling, as the model can correctly query colors without exact geometry. Finally, providing a complete supervision signal supports learning a well-structured latent space and enhances generalizability, leading to improved texture prediction quality, as demonstrated by our experiments in Sec. 4.5.2. Notably, although similar texture extrapolation methods have been discussed in [2], our work is the first to demonstrate that such a representation can serve as a strong training target to enhance texture generation, which constitutes one of our key contributions.

4. Experiments

4.1. Experimental Setup

In this section, we separately evaluate the generative ability of our overall system to demonstrate the effectiveness of our framework. (Detailed in Sec. 4.2) and the refinement ability of LTM to support our main claim (detailed in Sec. 4.3). Unlike previous methods [1, 51], we build two complementary benchmarks that jointly assess performance on both artist-created and generated meshes. For real (artist-created) meshes, we follow the evaluation criteria of MetaTexGen [1]. For generative meshes, we adopt Craftsman [22] and HY2.0 [51] to synthesize 3D objects for texturing. This setup allows a comprehensive evaluation of our model's texture generation performance under diverse, real-world scenarios. Due to the limited space, further experimental details are provided in the Supplementary.

4.2. Texture Generation Comparison

We first evaluate our method visually across meshes from different sources (handcrafted or generated). We compare against SoTA image-based texturing approaches, including Paint3D [44], MV-Adapter [15] and Hunyuan3D-Paint(2.0 [51] and 2.1 [12, 16]). We further provide comparisons with proprietary methods such as Tripo, Rodin and Meshy to further assess texturing quality in Supplementary. As demonstrated in Fig. 5, our model exhibits strong parsing capability and high geometric adherence, achieving overall superior fidelity. Specifically, in the first row (the mask example), under extreme lighting conditions, our model is still able to reconstruct a plausible albedo and produce textures that perfectly follow the geometry of the teeth. In the second column, our model successfully interprets the distinctive stripe-like ventilation pattern similar to window blinds and faithfully reproduces it, while all other methods fail. Moreover, our rendering of the rust surface appears significantly sharper and more natural. In the last column, when dealing with compositions of multiple small components, our model is also capable of accurate parsing and generation. In contrast, other methods either suffer from color bleeding, where white and green regions are confused at geometric boundaries. Moreover, HY2.0 and HY2.1 adopt rule-based inpainting methods, causing the occluded region (the leg) to retain noticeable artifacts, whereas our LTM successfully recovers the missing parts.

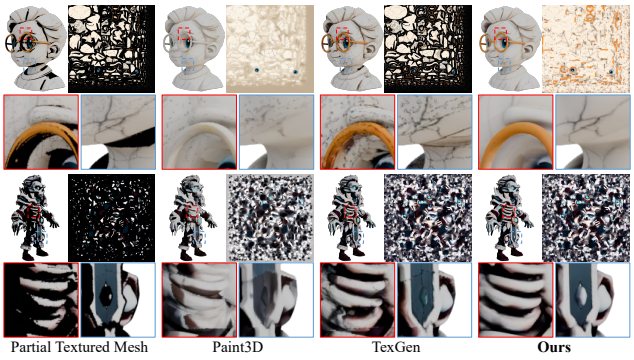


Figure 6. Qualitative comparison of refinement stage across different methods. Automatic unwrapping produces fragmented and noisy UV layouts on the face, which challenges UV-based methods like Paint3D and TexGen. In contrast, our method yields smooth, geometry-consistent textures (detailed analysis is provided in Sec. 4.3; best viewed by zooming in).

We also compare our method quantitatively with several metrics with *MVD-based models*: Hunyuan3D-Paint [51], TexPainter [46], Paint3D [44] and *Octree-based native 3D model*: TexGaussians [46]. For semantic similarity, we use CLIP-based FID (FID_{CLIP}) via Clean-FID [31], and CLIP-MMD (CMMD) [17] for finer fidelity assessment. CLIP-Score [52] measures alignment with input prompts, while LPIPS [48] evaluates perceptual similarity to ground truth. As shown in Tab. 1, our method outperforms the current state-of-the-art across both benchmarks. Moreover, TexGaussians—a single-stage native 3D generation model that uses rendering for supervision—performs poorly across the benchmarks compared with our method, especially on generative meshes that demand higher generalizability. These findings highlight two points. First, 3D-

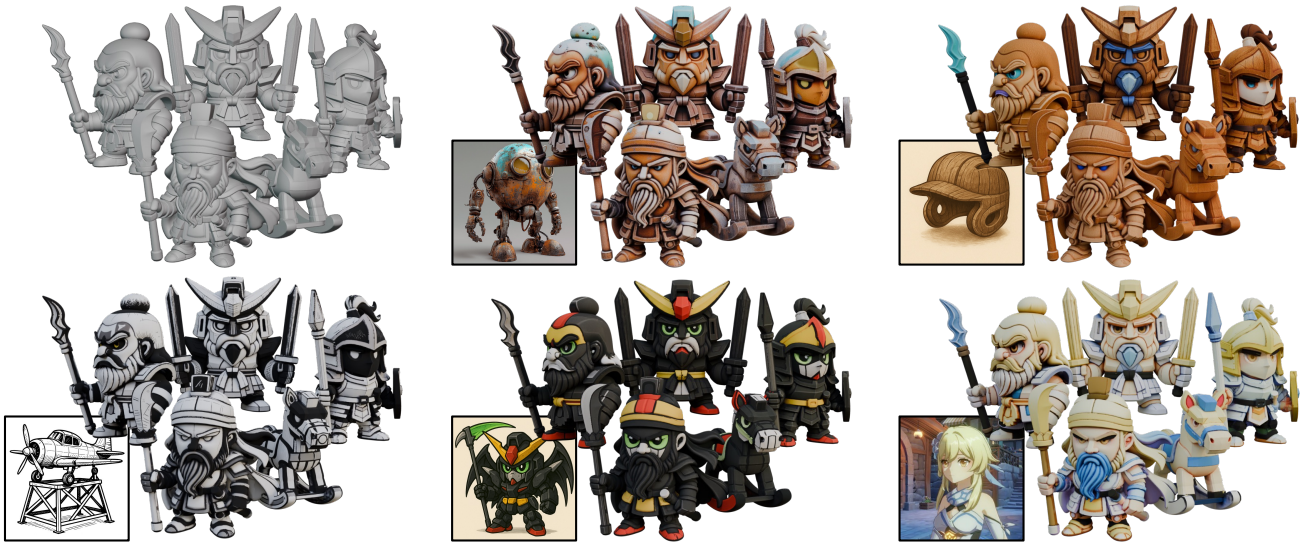


Figure 7. Qualitative results of stylized texturing. We evaluate our method with various input images (left-bottom side of each block), and it can robustly generate high-fidelity stylized textures (Mesh generated by Hunyuan2.5 [19]).

Table 2. Refinement Stage Comparison, $PSNR_{uv}{}^{*}$ indicates the PSNR calculated exclusively in the invisible regions, whereas $PSNR_{uv}$ is computed over the entire set of valid UV regions.

Methods.	$PSNR_{uv}\uparrow$	$PSNR_{uv}{}^{*}\uparrow$	PSNR↑	SSIM↑	LPIPS↓
Paint3D-UVpaint	18.61	15.22	23.90	0.971	0.042
TexGEN	20.47	17.07	25.19	0.977	0.040
Ours	**23.01**	**19.89**	**30.45**	**0.989**	**0.023**

native supervision is necessary. Second, native 3D texture approaches are currently better suited for refinement or inpainting, rather than as standalone generative solutions because of the scarcity of data. Therefore, leveraging 2D diffusion priors remains critical. Our method achieves highly competitive results across diverse models, demonstrating its robustness in fully texturing 3D shapes with varying topologies.

Since the metrics can only evaluate adherence to image/text instructions, we further incorporate a user study ("User-perf.") to assess the perceptual quality of textures generated on the generative meshes from Craftsman [22] (details of user study are provided in the Supplementary). As shown in the last column of Tab. 1, our method demonstrates significant advantages in human evaluations, indicating that it substantially outperforms previous state-of-the-art methods in texture quality.

4.3. Refinement Stage Comparison

In this section, we evaluate the effectiveness of the second stage to support our main claim — that refinement should be performed in 3D space rather than in UV space. Specifically, we take the generated images from the first stage as input and compare different approaches for texture refinement. We primarily evaluate the completeness and consistency of textures in self-occluded regions, which serve as a key indicator of the refinement quality.

We benchmark our approach against SoTA UV-based methods, including Paint3D [44] and TexGen [43]. As shown in Fig. 6, our method consistently achieves better results. In the first column, automatic UV unwrapping results in numerous small, fragmented regions on the face. UV-based methods struggle to refine these areas, leading to noticeable color inconsistencies and poor results. In contrast, our approach produces smooth and coherent textures. Furthermore, our model better respects the semantic consistency of texture sets. For example, in the first case, the glasses are seamlessly completed, and in the second case, both the ribs and the small emblem are accurately and coherently inpainted by our method.

We further perform a quantitative evaluation of our refinement stage on the artist-created mesh benchmark to rigorously assess the completeness and overall quality of the textures generated by our pipeline. In addition to standard visual quality metrics such as PSNR, SSIM [38], and LPIPS [48], we introduce an additional measure, $PSNR_{uv}$, which computes PSNR on the predicted surface using sampled UV points. This metric directly captures the model's ability to recover textures in self-occluded regions, providing a more faithful assessment of refinement performance. As shown in Tab. 2, the results demonstrate that our second stage achieves the best performance in preserving visible regions and completing occluded areas. These results suggest that our approach provides a more effective paradigm compared to UV-space operations or methods that treat texture

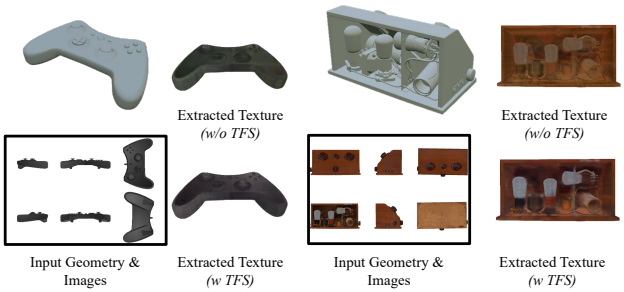


Figure 8. Visualization of the effectiveness of Texture Function Supervision (TFS). Under identical training iterations, models trained with TFS yield significantly higher-quality and completed textures compared to those supervised solely on surface signals (best viewed by zooming in).

Table 3. Ablation studies of drop training strategy in MV texture generation.

Methods.	MV Texture Generation					
	CMMD↓	FID_{CLIP}↓	$CLIP_{score}$↑	LPIPS↓	Mem. Cost↓	Speed↓
w/o DT	0.912	17.49	0.839	**0.087**	69.2GB	38.76s/it
w/ DT	**0.826**	**16.03**	**0.844**	0.090	**53.6GB**	**21.50s/it**

purely as a 2D image. This is because texture appearance is often closely tied to the underlying geometry, and operating in a 3D functional space allows our method to better incorporate geometric structure during refinement. Overall, the findings further support our claim that operating texture in 3D functional space s more suitable for representing texture compared to UV space.

4.4. Stylized Texturing

In this section, we demonstrate that our method can be applied to stylized texturing for 3D assets. Stylized texturing refers to the process of generating texture maps that not only align with the geometry of the 3D object but also reflect the visual style of a given reference image—such as oil painting, metallic finish, or cartoon shading. As illustrated in Fig. 7, our approach enables the creation of realistic and coherent textures while faithfully adapting to the appearance characteristics of different style exemplars.

4.5. Ablation Studies

In this section, we systematically evaluate all our proposed key components, including a comparison of the effectiveness of the patch dropout strategy (Sec. 4.5.1) and an analysis of the impact of texture function supervision (Sec. 4.5.2). Due to limited space, in addition to these two ablations that support our core contributions, we further provide additional experiments verifying the drop training strategy in normal estimation, as well as generalizability studies of our LTM, in the Supplementary. It is important to note that, due to limitations in computational resources, certain ablation experiments were performed using fewer training iterations than the full model.

Table 4. Ablation studies on Texture function. TF supervision can provide complete training for LTM.

Methods.	$PSNR_{uv}$↑	PSNR↑
w/o Texture Function Supervision	20.31	25.81
w Texture Function Supervision	**20.99**	**27.01**

4.5.1. Drop Training Strategy

In this section, we evaluate our proposed drop training strategy. We adopt the Artist-Created Mesh comparison setting described in Sec. 4.2. As shown in Tab. 3, our experiments demonstrate that using dropout during training achieves comparable performance in texture generation while significantly accelerating training. Specifically, dropping 50% of tokens in the multi-view texture generation task reduces memory usage by 22.5% and speeds up training by approximately 44.5% (evaluated on A800 with batch size 4). These results indicate that our drop training strategy serves as a useful plug-and-play approach for fine-tuning diffusion models.

4.5.2. Texture Functions Supervision

In this section, we evaluate the effectiveness of using texture functions (TFs) as supervision within our LTM framework. A straightforward alternative is to directly supervise the LTM using rendered images or surface-sampled RGB values, as adopted in prior works such as [18, 30, 42]. We compare this baseline (surface supervision) with our proposed TF supervision. It is worth noting that, unlike the setup in Tab. 2, this evaluation textures the entire model directly through LTM query points, without blending with partial texture geometry. This allows for a comprehensive assessment of the underlying representation's effectiveness.

As shown in Tab. 4 and Fig. 8, supervising with texture functions consistently yields superior performance, achieving the highest PSNR, which reflects improved fidelity and perceptual quality. In particular, the highest $PSNR_{uv}$ scores indicate a significant enhancement in texture completeness. These results demonstrate the effectiveness of our proposed TFs, which provide complete supervision for LTM.

5. Conclusion

We introduced UniTEX, a two-stage framework for high-fidelity 3D texture generation. By integrating a fine-tuned FLUX model with our drop-training strategy, and coupling the Large Texturing Model (LTM) with Texture Function supervision, UniTEX not only bypasses UV unwrapping and produces complete 3D textures, but also achieves high-fidelity appearance. Extensive experiments on both artist-created and generative 3D model benchmarks demonstrate that UniTEX delivers superior geometric consistency, perceptual quality, and scalability compared to existing texturing methods.

References

[1] Raphael Bensadoun, Yanir Kleiman, Idan Azuri, Omri Harosh, Andrea Vedaldi, Natalia Neverova, and Oran Gafni. Meta 3d texturegen: Fast and consistent texture generation for 3d objects. *arXiv preprint arXiv:2407.02430*, 2024. 1, 2, 3, 6

[2] Robert Bridson. *Fluid simulation for computer graphics*. AK Peters/CRC Press, 2015. 2, 5, 6

[3] Tianshi Cao, Karsten Kreis, Sanja Fidler, Nicholas Sharp, and Kangxue Yin. Texfusion: Synthesizing 3d textures with text-guided image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4169–4181, 2023. 3

[4] Dave Zhenyu Chen, Yawar Siddiqui, Hsin-Ying Lee, Sergey Tulyakov, and Matthias Nießner. Text2tex: Text-driven texture synthesis via diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 18558–18568, 2023. 3

[5] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In *ICCV*, 2023. 3

[6] Rui Chen, Jianfeng Zhang, Yixun Liang, Guan Luo, Weiyu Li, Jiarui Liu, Xiu Li, Xiaoxiao Long, Jiashi Feng, and Ping Tan. Dora: Sampling and benchmarking for 3d shape variational auto-encoders. *arXiv preprint arXiv:2412.17808*, 2024. 5

[7] Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. Longlora: Efficient fine-tuning of long-context large language models. *arXiv preprint arXiv:2309.12307*, 2023. 4

[8] Wei Cheng, Juncheng Mu, Xianfang Zeng, Xin Chen, Anqi Pang, Chi Zhang, Zhibin Wang, Bin Fu, Gang Yu, Ziwei Liu, and Liang Pan. Mvpaint: Synchronized multi-view diffusion for painting anything 3d. *arXiv preprint arxiv:2411.02336*, 2024. 1, 3

[9] Kangle Deng, Timothy Omernick, Alexander Weiss, Deva Ramanan, Jun-Yan Zhu, Tinghui Zhou, and Maneesh Agrawala. Flashtex: Fast relightable mesh texturing with lightcontrolnet. In *European Conference on Computer Vision*, pages 90–107. Springer, 2024. 3

[10] Linwei Dong, Qingnan Fan, Yihong Guo, Zhonghao Wang, Qi Zhang, Jinwei Chen, Yawei Luo, and Changqing Zou. Tsd-sr: One-step diffusion with target score distillation for real-world image super-resolution. *arXiv preprint arXiv:2411.18263*, 2024. 3

[11] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024. 1

[12] Yifei Feng, Mingxin Yang, Shuhui Yang, Sheng Zhang, Jiaao Yu, Zibo Zhao, Yuhong Liu, Jie Jiang, and Chunchao Guo. Romantex: Decoupling 3d-aware rotary positional embedded multi-attention network for texture synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17203–17213, 2025. 6

[13] Jingyu Guo, Sensen Gao, Jia-Wang Bian, Wanhu Sun, Heliang Zheng, Rongfei Jia, and Mingming Gong. Hyper3d: Efficient 3d representation via hybrid triplane and octree feature for enhanced 3d shape variational auto-encoders. *arXiv preprint arXiv:2503.10403*, 2025. 5

[14] 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

[15] Zehuan Huang, Yuan-Chen Guo, Haoran Wang, Ran Yi, Lizhuang Ma, Yan-Pei Cao, and Lu Sheng. Mv-adapter: Multi-view consistent image generation made easy. *arXiv preprint arXiv:2412.03632*, 2024. 6

[16] Team Hunyuan3D, Shuhui Yang, Mingxin Yang, Yifei Feng, Xin Huang, Sheng Zhang, Zebin He, Di Luo, Haolin Liu, Yunfei Zhao, et al. Hunyuan3d 2.1: From images to high-fidelity 3d assets with production-ready pbr material. *arXiv preprint arXiv:2506.15442*, 2025. 6

[17] Sadeep Jayasumana, Srikumar Ramalingam, Andreas Veit, Daniel Glasner, Ayan Chakrabarti, and Sanjiv Kumar. Rethinking fid: Towards a better evaluation metric for image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9307–9315, 2024. 6

[18] Lutao Jiang, Jiantao Lin, Kanghao Chen, Wenhang Ge, Xin Yang, Yifan Jiang, Yuanhuiyi Lyu, Xu Zheng, and Yingcong Chen. Dimer: Disentangled mesh reconstruction model. *arXiv preprint arXiv:2504.17670*, 2025. 5, 8

[19] Zeqiang Lai, Yunfei Zhao, Haolin Liu, Zibo Zhao, Qingxiang Lin, Huiwen Shi, Xianghui Yang, Mingxin Yang, Shuhui Yang, Yifei Feng, et al. Hunyuan3d 2.5: Towards high-fidelity 3d assets generation with ultimate details. *arXiv preprint arXiv:2506.16504*, 2025. 7

[20] Sixu Li, Chaojian Li, Wenbo Zhu, Boyang Yu, Yang Zhao, Cheng Wan, Haoran You, Huihong Shi, and Yingyan Lin. Instant-3d: Instant neural radiance field training towards on-device ar/vr 3d reconstruction. In *Proceedings of the 50th Annual International Symposium on Computer Architecture*, pages 1–13, 2023. 3, 5

[21] Weiyu Li, Rui Chen, Xuelin Chen, and Ping Tan. Sweetdreamer: Aligning geometric priors in 2d diffusion for consistent text-to-3d. *arXiv preprint arXiv:2310.02596*, 2023. 3

[22] Weiyu Li, Jiarui Liu, Rui Chen, Yixun Liang, Xuelin Chen, Ping Tan, and Xiaoxiao Long. Craftsman: High-fidelity mesh generation with 3d native generation and interactive geometry refiner. *arXiv preprint arXiv:2405.14979*, 2024. 2, 5, 6, 7

[23] Yangguang Li, Zi-Xin Zou, Zexiang Liu, Dehu Wang, Yuan Liang, Zhipeng Yu, Xingchao Liu, Yuan-Chen Guo, Ding Liang, Wanli Ouyang, et al. Triposg: High-fidelity 3d shape synthesis using large-scale rectified flow models. *arXiv preprint arXiv:2502.06608*, 2025. 2, 5

[24] Yixun Liang, Weiyu Li, Rui Chen, Fei-Peng Tian, Jiarui Liu, Ying-Cong Chen, Ping Tan, and Xiao-Xiao Long. Iris3d: 3d generation via synchronized diffusion distillation. *ACM Trans. Graph.*, 45(1), 2025. 2

[25] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *CVPR*, 2023. 3

[26] Jialun Liu, Chenming Wu, Xinqi Liu, Xing Liu, Jinbo Wu, Haotian Peng, Chen Zhao, Haocheng Feng, Jingtuo Liu, and Errui Ding. Texoct: Generating textures of 3d models with octree-based diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4284–4293, 2024. 3

[27] Yuxin Liu, Minshan Xie, Hanyuan Liu, and Tien-Tsin Wong. Text-guided texturing by synchronized multi-view diffusion. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–11, 2024. 1, 3

[28] 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. 2

[29] Kunming Luo, Hongyu Yan, Yuan Liu, Zihao Zhang, Manyuan Zhang, Wenping Wang, and Ping Tan. Ctr3d: Cross-view token reduction for dense multi-view generation. In *Thirteenth International Conference on 3D Vision*. 2

[30] Michael Oechsle, Lars Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger. Texture fields: Learning texture representations in function space. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4531–4540, 2019. 3, 5, 8

[31] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On aliased resizing and surprising subtleties in gan evaluation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11410–11420, 2022. 6

[32] Christian Reiser, Rick Szeliski, Dor Verbin, Pratul Srinivasan, Ben Mildenhall, Andreas Geiger, Jon Barron, and Peter Hedman. Merf: Memory-efficient radiance fields for real-time view synthesis in unbounded scenes. *ACM Transactions on Graphics (TOG)*, 42(4):1–12, 2023. 5

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

[34] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 1, 3

[35] Yawar Siddiqui, Justus Thies, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. Texturify: Generating textures on 3d shape surfaces. In *European Conference on Computer Vision*, pages 72–88. Springer, 2022. 3, 5

[36] Zhenxiong Tan, Songhua Liu, Xingyi Yang, Qiaochu Xue, and Xinchao Wang. Ominicontrol: Minimal and universal control for diffusion transformer. *arXiv preprint arXiv:2411.15098*, 2024. 3

[37] Shitao Tang, Jiacheng Chen, Dilin Wang, Chengzhou Tang, Fuyang Zhang, Yuchen Fan, Vikas Chandra, Yasutaka Furukawa, and Rakesh Ranjan. Mvdiffusion++: A dense high-resolution multi-view diffusion model for single or sparse-view 3d object reconstruction. In *European Conference on Computer Vision*, pages 175–191. Springer, 2024. 4

[38] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *TIP*, 2004. 7

[39] 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. 3, 5

[40] Bojun Xiong, Jialun Liu, Jiakui Hu, Chenming Wu, Jinbo Wu, Xing Liu, Chen Zhao, Errui Ding, and Zhouhui Lian. Texgaussian: Generating high-quality pbr material via octree-based 3d gaussian splatting. *arXiv preprint arXiv:2411.19654*, 2024. 3

[41] Hongyu Yan, Kunming Luo, Weiyu Li, Yixun Liang, Shengming Li, Jingwei Huang, Chunchao Guo, and Ping Tan. Posemaster: Generating 3d characters in arbitrary poses from a single image. *arXiv preprint arXiv:2506.21076*, 2025. 2

[42] Xin Yu, Peng Dai, Wenbo Li, Lan Ma, Zhengzhe Liu, and Xiaojuan Qi. Texture generation on 3d meshes with point-uv diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4206–4216, 2023. 3, 8

[43] Xin Yu, Ze Yuan, Yuan-Chen Guo, Ying-Tian Liu, Jianhui Liu, Yangguang Li, Yan-Pei Cao, Ding Liang, and Xiaojuan Qi. Texgen: a generative diffusion model for mesh textures. *ACM Transactions on Graphics (TOG)*, 43(6):1–14, 2024. 2, 3, 7

[44] Xianfang Zeng, Xin Chen, Zhongqi Qi, Wen Liu, Zibo Zhao, Zhibin Wang, Bin Fu, Yong Liu, and Gang Yu. Paint3d: Paint anything 3d with lighting-less texture diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4252–4262, 2024. 1, 2, 3, 6, 7

[45] Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Transactions on Graphics (TOG)*, 42(4):1–16, 2023. 5

[46] Hongkun Zhang, Zherong Pan, Congyi Zhang, Lifeng Zhu, and Xifeng Gao. Texpainter: Generative mesh texturing with multi-view consistency. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 3, 6

[47] 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. *ACM Transactions on Graphics (TOG)*, 43(4):1–20, 2024. 3

[48] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 6, 7

[49] Yuqing Zhang, Yuan Liu, Zhiyu Xie, Lei Yang, Zhongyuan Liu, Mengzhou Yang, Runze Zhang, Qilong Kou, Cheng Lin, Wenping Wang, et al. Dreammat: High-quality pbr material generation with geometry-and light-aware diffusion models. *ACM Transactions on Graphics (TOG)*, 43(4):1–18, 2024. 3

[50] Yuxuan Zhang, Yirui Yuan, Yiren Song, Haofan Wang, and Jiaming Liu. Easycontrol: Adding efficient and flexible control for diffusion transformer. *arXiv preprint arXiv:2503.07027*, 2025. 3

[51] Zibo Zhao, Zeqiang Lai, Qingxiang Lin, Yunfei Zhao, Haolin Liu, Shuhui Yang, Yifei Feng, Mingxin Yang, Sheng Zhang, Xianghui Yang, et al. Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation. *arXiv preprint arXiv:2501.12202*, 2025. 2, 3, 5, 6

[52] SUN Zhengwentai. clip-score: CLIP Score for PyTorch. https://github.com/taited/clip-score, 2023. Version 0.2.1. 6