

TexPainter: Generative Mesh Texturing with Multi-view Consistency

Hongkun Zhang
H.K.Zhang5813@gmail.com
Southeast University
Nanjing, Jiangsu, China

Zherong Pan
zrpan@global.tencent.com
LightSpeed Studios
Seattle, WA, USA

Congyi Zhang
congyiz@cs.ubc.ca
The University of British Columbia
Vancouver, BC, Canada
TransGP & HKU, Hong Kong

Lifeng Zhu*
lfzhulf@gmail.com
Southeast University
Nanjing, Jiangsu, China

Xifeng Gao
xifgao@global.tencent.com
LightSpeed Studios
Seattle, WA, USA



Albert Einstein, full color

a Canon AT-1 Retro camera

Mandalorian helmet, Star War

wooden shield adorned with iron embellishments

Figure 1: We have evaluated our method on a row of 3D models and our method consistently generates high-quality texture images with multi-view consistency, some of which are illustrated with two views for each model, along with the text prompt below.

ABSTRACT

The recent success of pre-trained diffusion models unlocks the possibility of the automatic generation of textures for arbitrary 3D meshes in the wild. However, these models are trained in the screen space, while converting them to a multi-view consistent texture image poses a major obstacle to the output quality. In this

paper, we propose a novel method to enforce multi-view consistency. Our method is based on the observation that latent space in a pre-trained diffusion model is noised separately for each camera view, making it difficult to achieve multi-view consistency by directly manipulating the latent codes. Based on the celebrated Denoising Diffusion Implicit Models (DDIM) scheme, we propose to use an optimization-based color-fusion to enforce consistency and indirectly modify the latent codes by gradient back-propagation. Our method further relaxes the sequential dependency assumption among the camera views. By evaluating on a series of general 3D models, we find our simple approach improves consistency and overall quality of the generated textures as compared to competing state-of-the-arts. Our implementation is available at: <https://github.com/Quantuman134/TexPainter>

*Corresponding author

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-XXXX-X/18/06...\$15.00
<https://doi.org/10.1145/3592143>

CCS CONCEPTS

• **Computing methodologies** → **Mesh models**.

KEYWORDS

Text-guided Texture Generation, Diffusion Model

ACM Reference Format:

Hongkun Zhang, Zherong Pan, Congyi Zhang, Lifeng Zhu, and Xifeng Gao. 2023. TexPainter: Generative Mesh Texturing with Multi-view Consistency. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, Article 1, 11 pages. <https://doi.org/10.1145/3592143>

1 INTRODUCTION

Automatic content creation is one of the ultimate goals of computer graphics. Decades of efforts, including conventional content creation techniques [Chen et al. 2008; Müller et al. 2006; Prusinkiewicz et al. 1996], have been invested in this domain. The most recent success of Latent Diffusion Model (LDM) [Rombach et al. 2022] trained on large-scale internet image dataset significantly advances the multi-model expressivity of generative models. Since then, continued efforts have been made to extend the expressivity from 2D images to 3D models. However, due to the limited dataset and computational resources, 3D diffusion models [Vahdat et al. 2022; Zhou et al. 2021] still cannot achieve similar variety and scalability as their 2D counterpart. In parallel, researchers have turned to extracting 3D information from pre-trained 2D LDM. An exemplary technique is DreamFusion [Poole et al. 2023], which optimizes a neural radiance field using Score Distillation Sampling (SDS). This technique, however, tends to produce appearances with unnatural colors for the geometry. To bridge this gap, the latest texture generation techniques [Cao et al. 2023; Chen et al. 2023c; Richardson et al. 2023] propose to distill texture images for given 3D models from depth-conditioned, pre-trained 2D LDM.

Our work aims at extracting texture images of a consistently higher quality from pre-trained 2D LDM. Our key technical challenge lies in resolving the multi-view consistency, i.e., the texture should lead to semantically and visually consistent rendered appearance across all camera views. Regretfully, existing techniques [Cao et al. 2023; Chen et al. 2023c; Richardson et al. 2023] still cannot achieve satisfactory results due to several reasons. First, the early approaches [Chen et al. 2023c; Richardson et al. 2023] runs the denoising process for each view, sequentially, leading to sub-optimal results. Later, Cao et al. [2023] tackle this problem by tightly coupling the denoising process with multi-view fusion. They choose to fuse the texture in latent space using noisy screen-space multi-view images. However, since latent codes for different views are noised in separate diffusion processes, such manipulation can be detrimental to the image quality and even counteract the consistency. Further, these methods introduce unnecessary assumptions between multiple views. For example, Chen et al. [2023c]; Richardson et al. [2023] assume the next camera view can either keep, refine, or overwrite the texture generated from the previous camera view. While [Cao et al. 2023] assumes the rendered images from multiple views have sequential dependence. This assumption is counter-intuitive, because difference camera views are not directly correlated through a common texture image.

We propose a new approach for generating a multi-view consistent texture image from a given 3D model and a text prompt. As the key point of departure from prior works, our method avoids direct operations on the latent codes and does not use assumptions on inter-view correlation. Instead, we fuse multiple views by joint optimization of the latents. Our technique is based on the mechanism of the celebrated DDIM scheme [Song et al. 2021a]. During each denoising step, DDIM estimates the ultimate noiseless latent state to predict the next noise level. We then decode these noiseless latents into color space and blend them to form a texture image. We choose to update the DDIM latents across all views by optimization, such that they generate the same image as one rendered using the blended texture image. As such, we also eliminate the aforementioned assumption on sequential dependence. Extended experiments and user study confirms that our simple approach achieves consistent improvements on texture quality.

2 RELATED WORK

We review related work on the automated generation of 3D contents, including both geometry and appearance.

2.1 Classical 2D/3D Content Generation

In the early stage, content generation techniques are generally confined to a specific category of geometry and appearance models. For example, texture synthesize methods [Wei et al. 2009] extend a small exemplary texture tile to larger textures. Texture transfer methods [Mertens et al. 2006] replicate the texture across different 3D models. [Chen et al. 2022b] introduces a network designed to transfer texture from an example. Procedural methods can generate both 2D textures [Dong et al. 2020] and 3D geometries [Smelik et al. 2014]. For certain categories of models, such as trees [Sun et al. 2009] and buildings [Bao et al. 2013; Talton et al. 2011], or branch structures [Sibbing et al. 2010], specialized techniques can be designed to generate both geometry and appearance with significant varieties. However, all these techniques potentially suffer from two common drawbacks. First, these methods can only generate contents varied in low-level details, while the high level semantics must be kept fixed. Second, they require considerable domain knowledge to use, leading to a non-trivial learning curve.

2.2 Learnable 3D Generative Models

Deep learning techniques have been applied to content generation and achieved significant success in the past decade. Their success is backed by the everlasting efforts to search for powerful generative models that can efficiently represent complex multi-model distributions, of which two representative models are Variational AutoEncoders (VAE) and Generative Adversarial Networks (GAN). While originally experimented on 2D image datasets, they have been extended in [Brock et al. 2016; Wu et al. 2016] to represent 3D geometry via voxel grid representation. However, these works are focused on generating only 3D geometry, instead of full appearance models. This gap is bridged by several follow-up works that synthesize appearance models. For example, Text2Mesh [Michel et al. 2022] infers stylish mesh textures and displacements to match a given text prompt. Texture fields [Oechsle et al. 2019] and Texturify [Siddiqui et al. 2022] generate the texture for a given 3D

mesh to match the appearance of a 2D image. They use VAE to encode the input cues and GAN-loss to optimize the texture. Yu et al. [2021] and Mesh2Tex [Bokhovkin et al. 2023] achieve decorating of existing 3D shapes by training a conditional texture generator, leveraging GAN. While the aforementioned techniques are focused on generating the color field, TANGO [Chen et al. 2022a] goes beyond this paradigm to generate a complete BRDF model for an existing 3D object, where the generated BRDF model is matched with a text prompt using the CLIP loss. Unlike all these works that generate appearance for given geometry, GET3D [Gao et al. 2022] jointly generate textured meshes. They parametrically represent the geometry and color through auto-encoding, and use GAN to train the joint generative model. Similarly, ShaDDR [Chen et al. 2023b] construct both the geometry and texture from a provided example drawing inspiration from GAN. Compared with the more recent LDM, VAE and GAN has limited expressivity. As a result, separate models oftentimes need to be trained for each category of data, which limits their domain of usage.

2.3 2D/3D Diffusion Models

Diffusion models [Ho et al. 2020; Song et al. 2021b] demonstrate significantly improved ability to approximate complex distributions. Based on this model, there are many advances in the area of 2D image generation over the past two years. In particular, the celebrated LDM [Rombach et al. 2022] produces high-quality images with affordable memory and computation. In addition, the multi-model conditioning of LDM, using text and depth images, significantly improves their amenability to non-expert users. A key technique behind the conditional generation is the blending between classified and classifier-free guidance [Ho and Salimans 2021]. In parallel, a series of methods are designed to accelerate the sampling of the reverse diffusion process [Lu et al. 2022; Song et al. 2021a].

Extending from 2D to 3D content generation, there are several attempts that train diffusion models to directly generate 3D assets [Karnewar et al. 2023; Müller et al. 2023; Zeng et al. 2022]. For appearance generation, in particular, Point-UV [Yu et al. 2023a] is a 3D diffusion model that predicts the color of a given mesh and synthesises the texture. However, the variety and quality of these generated 3D contents are considerably lower than their 2D counterparts [Rombach et al. 2022]. This is largely because 3D content generation requires models with considerably more parameters, while the amount of computational resources and available datasets are severely inadequate.

2.4 3D Content Distillation from 2D LDM

Considering the inherent difficulty to train full-fledged large 3D LDM, researchers have considered extracting 3D contents from pre-trained 2D diffusion model [von Platen et al. 2022]. The seminal work of DreamFusion [Poole et al. 2023] proposes SDS to optimize a neural radiance field [Mildenhall et al. 2021] from pre-trained models. Through the optimization, DreamFusion generates colored geometry conforming to the input text prompt. There are also other SDS-based methods with further improved generation quality on geometry and/or appearance [Chen et al. 2023a; Lin et al. 2023; Metzger et al. 2023]. For higher quality or better alignment with input condition, several methods even fine-tuned the pre-trained LDM [Wang et al. 2023; Yu et al. 2023b]. However, an noticeable

limitation of SDS is the excessive usage of classifier-free guidance weighting [Ho and Salimans 2021], which suffers from a low fidelity with over-saturation and overexposure. Besides, DG3D [Zuo et al. 2023] also incorporates a diffusion model to produce textured meshes, although the appearance quality remains constrained.

Instead of generating both geometry and appearance all at once, the depth-conditioned LDM enables the application of generating appearance for a given geometry. To conquer this seemingly easier task, the major technical challenge lies in multi-view consistency, i.e., the appearance must be consistent across all possible camera views. To enforce such consistency, an intuitive idea is to generate images from a set of sampled views and then “paint” them onto the 3D model. Richardson et al. [2023] and Chen et al. [2023c] propose their texture generation methods based on this idea, where each new camera view revises the overlapping parts of the texture image from the previous view. However, this method still suffers from various artifacts including seam, noise, and meaningless fragments. Several texture painting works enhance consistency within their respective contexts. Paint3D [Zeng et al. 2023] trains a dedicated model to fill incomplete areas during the texture painting. MVDiffusion [Tang et al. 2023] focuses on panorama generation with a given mesh and fine-tune a 2D diffusion model to maintain consistency. To alleviate the inconsistency more directly, Cao et al. [2023] propose to sequentially correlate the noised images from all different views. This method operates entirely in the latent space and finally use an additional neural field optimization to reconstruct the color-space texture image. In our experiments, however, their latent-space manipulations can reduce the quality of the texture image. By comparison, we adopt the multi-view fusion in the color space and then solve an optimization during each denoising step to update the latents, which further eliminates the assumption on sequential inter-view correlation.

3 PRELIMINARIES

3.1 The Texture Painting Problem

Our intended problem takes the same form as prior works [Cao et al. 2023; Chen et al. 2023c; Richardson et al. 2023]. We are given a 3D object represented by a mesh $\mathcal{M} = \langle \mathcal{V}, \mathcal{F} \rangle$, with \mathcal{V} and \mathcal{F} being a set of vertices and facets, respectively. We further assume the appearance of the mesh is determined by a texture image and a UV map. The UV map defines an almost everywhere injective function $\mathcal{T} : \mathbb{R}^2 \mapsto \mathbb{R}^3$ mapping a point on the 2D plane to that of the 3D surface on \mathcal{M} , which assigns the texture color $I(u)$ to the surface point $\mathcal{T}(u)$. In addition to the surface mesh \mathcal{M} , we further assume users provide a text-based description of the mesh semantics and appearance, which is converted to a text prompt embedding h as the conditional input. Given \mathcal{M} and h , our goal is to automatically infer the texture image I that has consistent appearance and semantic meaning under arbitrary camera views.

3.2 Diffusion Model and Denoising Procedures

Our method is based on the pre-trained, depth-conditioned LDM [Rombach et al. 2022] for generating 2D images, guided by a text prompt. The diffusion model [Sohl-Dickstein et al. 2015] is a generative model inspired by thermal dynamics. First, we define the diffusion process that gradually injects Gaussian noise

into the data distribution $z_0 \sim p(z_0)$, leading to the following forward Markovian model $p(z_{1:T}|z_0) = \prod_{t=0}^{T-1} p(z_{t+1}|z_t)$ with $p(z_{t+1}|z_t) = \mathcal{N}\left(\sqrt{\frac{\alpha_t}{\alpha_{t-1}}}z_t, \left(1 - \frac{\alpha_t}{\alpha_{t-1}}\right)I\right)$ and $\alpha_t \in (0, 1]$ being a decreasing schedule of noise coefficients. In the forward process, the data distribution is blended into a pure Gaussian noise at the t th timestep. The diffusion model works by learning the inverse of the diffusion process that gradually removes noise from z_t . Taking the original Denoising Diffusion Probabilistic Models (DDPM) [Ho et al. 2020] for example, the noise component in z_t is predicted using a neural network $\epsilon_\theta(z_t, t, h)$, and we can sample the next level of less noisy version z_{t-1} by:

$$z_{t-1} \sim \mathcal{N}\left(\frac{1}{\sqrt{\alpha_t}}\left(z_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\epsilon_\theta(z_t, t, h)\right), \sigma_t I\right), \quad (1)$$

with $\beta_t = 1 - \alpha_t$, $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s$, and σ_t being the noise variance at level t . Here h is some latent code encoding the user-input guidance information. The original DDPM has been improved in its expressivity and inference efficacy in various prior works, of which the most outstanding results is LDM [Rombach et al. 2022] that propose to perform the inverse process in the latent space. We use their notation and denote the latent code as z and the original image as x . The pre-trained autoencoder is denoted as: $z = \mathcal{E}(x)$ and $x = \mathcal{D}(z)$. In parallel, DDPM suffers from slow inference due to its temporally sequential nature of the forward process. In view of this, the more recent DDIM [Song et al. 2021a] proposes to improve the inference cost by using a non-Markovian inverse process. The denoising step of DDIM in Equation 1 is replaced with:

$$z_{t-1} \sim \mathcal{N}\left(\sqrt{\alpha_{t-1}}\hat{z}_{0,t} + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\epsilon_\theta(z_t, t, h), \sigma_t I\right) \\ \hat{z}_{0,t} \triangleq \frac{z_t - \sqrt{1-\alpha_t}\epsilon_\theta(z_t, t, h)}{\sqrt{\alpha_t}}, \quad (2)$$

where we use $\hat{z}_{0,t}$ to denote the predicted noiseless data distribution at t th timestep. We will use this property to design our algorithm to enforce multi-view consistency.

3.3 Multi-view Inconsistency in Texture Painting

We detail the texture generation process to infer I from \mathcal{M} and h . During painting, we query the 2D LDM using text prompt h from a set \mathcal{C} of sampled camera view positions denoted as $\mathcal{C} = \{c^i\}$, where we use superscript to denote camera indices throughout. We denote $\mathcal{R}(\mathcal{M}, I, c)$ as a rendering function, and $\mathcal{R}_z, \mathcal{R}_x$ as renderers working in the latent and color space, respectively. Similarly, we denote I_z and I_x as the texture image in the corresponding space.

To solve the texture painting problem, prior methods run a separate denoising process for each c^i using LDM, generating a sequence of latent images z_t^i . We denote the updated latent texture using the first l camera views as I_z^l , then all these prior works assume a sequential dependence on camera views, i.e., the following probabilistic model:

$$p(I_{z,x}^j) = p(I_{z,x}^0) \prod_{k=1}^j p(I_{z,x}^k | I_{z,x}^{k-1}), \quad (3)$$

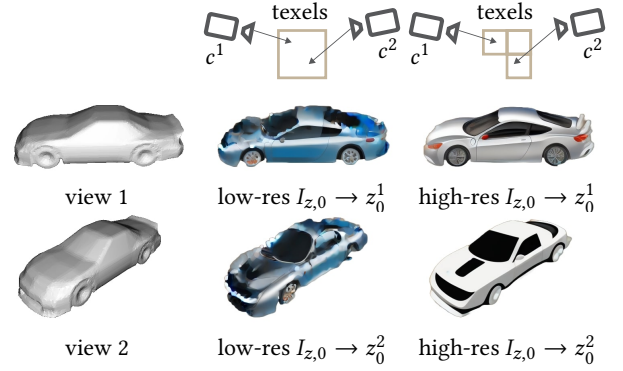


Figure 2: We run diffusion processes from two nearby views and enforce consistency by blending the noisy latent code into I_z during each step. Under a low-res I_z , two views are correlated by sampling largely the same set of texels, thus achieving multi-view consistency, but low-res I_z leads to low-quality blurry images (middle). Instead, clear images are derived under a high-res I_z , but the two views can fetch entirely different sets of texels due to the nearest sampling scheme used by SIMS, failing to achieve consistency (right).

either in latent or color space, as denoted by subscript. The first category of methods [Cao et al. 2023; Chen et al. 2023c] run the denoising process for each view and update the color texture sequentially, i.e., assuming subscript x in Equation 3. Later, Cao et al. [2023] noticed that sequential denoising leads to various artifacts. Instead, they proposed to couple the denoising process with texture fusion by sequentially merging noisy images into a latent texture during each step, i.e., assuming subscript z in Equation 3. To mitigate the inconsistency across multi-views in each denoising step, they use a meticulously designed update rule named SIMS. Unfortunately, as illustrated in Figure 2, we show that this method can still result in blurred images under a low-resolution I_z or inconsistent images using a high-resolution I_z , leading to a dilemma in choosing appropriate texture resolution.

4 METHOD

Our texture generator pipeline is illustrated in Figure 3, which is a modified multi-DDIM procedure [Song et al. 2021a] that enforces multi-view consistency. In this section, we detail our modified DDIM procedure in Section 4.1. Then, we propose two extensions to our pipeline in Section 4.2.

4.1 DDIM with Multi-view Consistency

To circumvent the pitfall of prior texture painting methods, we make two observations. First, although z_t^i during an intermediary denoising step has drastically different noise component across views, the $\hat{z}_{0,t}^i$ predicted by the DDIM scheme is an estimation at the 0th timestep, which is supposed to be noiseless. Second, we notice that the auto-encoder used by LDM [Rombach et al. 2022] can be highly nonlinear map, and consistency in the color space does not imply that in the latent space. Therefore, we propose to enforce multi-view consistency by modifying $\hat{z}_{0,t}^i$ in the color space. As illustrated in Figure 3, specifically, we perform denoising process for all the sampled views in parallel. For the t th timestep,

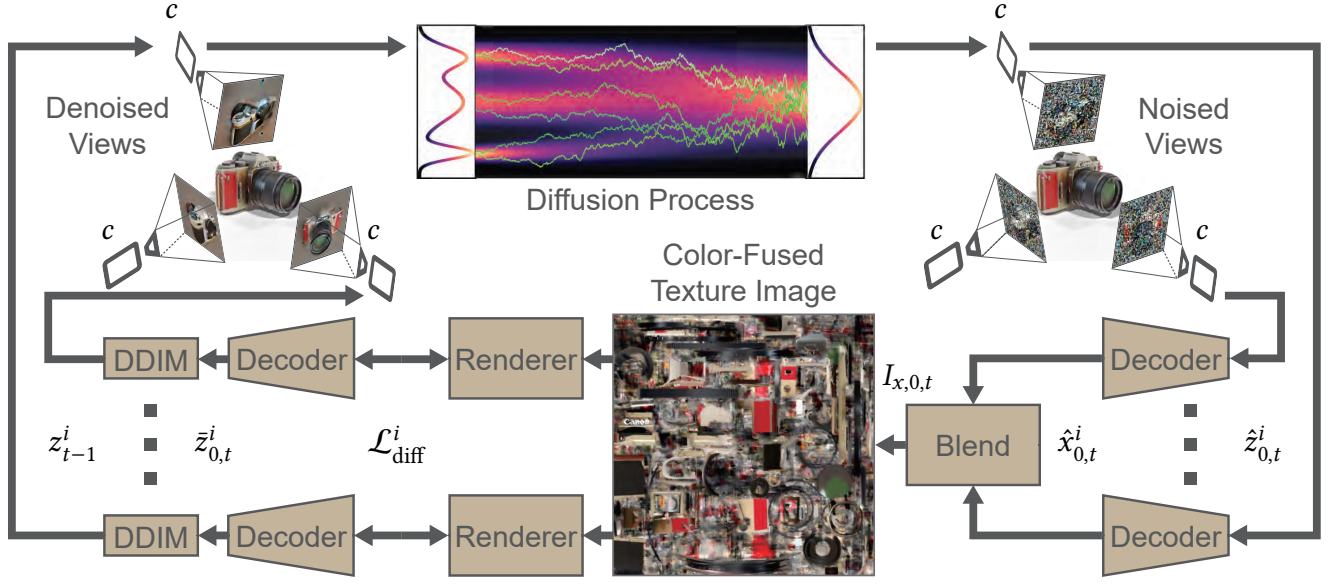


Figure 3: Our modified multi-DDIM procedure that enforces multi-view consistency. Each view runs a separate denoising procedure using DDIM scheme. For each denoising step, DDIM predicts a latent code $\hat{z}_{0,t}^i$ for the i th view at 0th timestep. These $\hat{z}_{0,t}^i$ are decoded to the color space, yielding $\hat{x}_{0,t}^i$. We then blend these views into a common color-space texture image by weighted averaging. Next, we perform an optimization to update $\hat{z}_{0,t}^i$ into $\bar{z}_{0,t}^i$ for all views, such that their decoded images match their corresponding rendered views using the blended texture image. These updated latent codes are then plugged into DDIM to predict the next noise level.

our method yields the set of latent images $\{z_t^i\}$. We then predict 0th timestep using Equation 2 to yield $\{\hat{z}_{0,t}^i\}$.

Next, we switch from latent to color space by applying the decoder, yielding predicted 0th timestep color images: $\hat{x}_{0,t}^i = \mathcal{D}(\hat{z}_{0,t}^i)$. In the color space, we then fuse the images to a color texture $I_{x,0,t}$. To this end, we use simple weighted averaging scheme. Specifically each pixel u lies on some facet with its outer-normal denoted as $n(u)$. The camera direction from i th view to $\mathcal{T}(u)$ is denoted as $c^i - \mathcal{T}(u)$. We assign the following weight to the i th view:

$$w^i = \max\left(0, \left\langle \frac{c^i - \mathcal{T}(u)}{\|c^i - \mathcal{T}(u)\|}, n(u) \right\rangle\right),$$

which assigns larger weights to views that are nearly orthogonal to the facet and cover more pixels. We then assign the color texture via the following scheme:

$$I_{x,0,t}(u) = \sum_{i \in \mathcal{C}(\mathcal{T}(u))} w^i \hat{x}_{0,t}^i(\mathcal{T}(u)) / \sum_{i=1}^{|\mathcal{C}|} w^i, \quad (4)$$

where we use $\mathcal{C}(\mathcal{T}(u))$ to denote all the cameras that are visible from $\mathcal{T}(u)$. Here we slightly abuse notation to use $\hat{x}_{0,t}^i(\mathcal{T}(u))$ to denote the sampling operator that fetches the pixel corresponding to world space coordinate $\mathcal{T}(u)$. Following [Cao et al. 2023], we use nearest neighbor scheme to perform the pixel interpolation.

With the color-fused texture image $I_{x,0,t}$, we now switch back from color to latent space, updating $\{\hat{z}_{0,t}^i\}$ to yield a set of adjusted latent codes $\{\bar{z}_{0,t}^i\}$ with multi-view consistency. To this end, we use an optimization to enforce that the updated $\{\bar{z}_{0,t}^i\}$ yield consistent images under corresponding views. Our optimization takes the

Algorithm 1 TexPainter

Input: Mesh \mathcal{M} , text prompt h
Output: Color texture image I
 Sample $\{z_T^i \sim \mathcal{N}(0, I)\}$
for $t = T, \dots, 0$ **do**
 Predict $\{\epsilon_\theta(z_t^i, t, h)\}$ using depth-conditioned LDM
 Predict $\{\hat{z}_{0,t}^i\}$ (Equation 2)
 Decode $\{\hat{x}_{0,t}^i = \mathcal{D}(\hat{z}_{0,t}^i)\}$
 Color-space fuse $I_{x,0,t}$ (Equation 4)
 if $t = 0$ **then**
 Return $I_{x,0,0}$
 end if
 Optimize $\{\bar{z}_{0,t}^i\}$ (Equation 5)
 Apply DDIM to yield $\{z_{t-1}^i\}$ (Equation 6)
end for

following form:

$$\operatorname{argmin}_{\{\bar{z}_{0,t}^i\}} \sum_{i=1}^{|\mathcal{C}|} \mathcal{L}_{\text{diff}}(\bar{z}_{0,t}^i, c^i) \quad (5)$$

$$\mathcal{L}_{\text{diff}}(z, c) \triangleq \|\mathcal{D}(z) - \mathcal{R}(\mathcal{M}, I_{x,0,t}, c)\|_1,$$

where we use $\{\hat{z}_{0,t}^i\}$ as our initial guess. Note that the rendering function in this optimization can be pre-evaluated before optimization. Therefore, the optimization does not require a differentiable renderer as used in [Cao et al. 2023] and only involves the derivatives of \mathcal{D} , which is relatively efficient to compute. Note that Equation 5 implies all camera views are treated equally without sequen-



Figure 4: We highlight the benefits of our joint optimization Equation 7 (left) as compared with Equation 4 (right). The joint optimization achieves better texture quality in areas not well-sampled by camera views, but increases the inference cost from 25min to 66min.

tial dependency. Finally, we plug the updated $\{\hat{z}_{0,t}^i\}$ into DDIM, obtaining:

$$z_{t-1}^i \sim \mathcal{N}\left(\sqrt{\alpha_{t-1}}z_{0,t}^i + \sqrt{1 - \alpha_{t-1} - \sigma_t^2 \epsilon_\theta(z_t^i, t, h)}, \sigma_t I\right). \quad (6)$$

We summarize our method in Algorithm 1, where our color-fusion procedure is highlighted in brown. As a computational drawback, our method requires solving the optimization Equation 5 per denoising step, while prior work [Cao et al. 2023] only applies the optimization once to reconstruct the color texture at last. Fortunately, since our objective function only requires back-propagation through the decoder \mathcal{D} , which is quite efficient, the cost of optimization is still manageable.

4.2 Extensions

We discuss two extensions to our method. First, we notice that using simple weighted averaging as in Equation 4 for color-fusion is a mere compromise between inference speed and texture quality. Our ultimate goal for this step is to adjust the predicted, noiseless latent codes such that they can be generated by a single unified texture image. We could achieve this by a joint optimization of $I_{x,0,t}$ and $\{\hat{z}_{0,t}^i\}$ via the following formulation:

$$\operatorname{argmin}_{\{\hat{z}_{0,t}^i\}, I_{x,0,t}} \sum_{i=1}^{|C|} \mathcal{L}_{\text{diff}}(\hat{z}_{0,t}^i, c^i) + \mathcal{L}_{\text{diff}}(\hat{z}_{0,t}^i, c^i). \quad (7)$$

In practice, we can still use $\{\hat{z}_{0,t}^i\}$ and Equation 4 as our initial guess for $\{\hat{z}_{0,t}^i\}$ and $I_{x,0,t}$, respectively, and then optimize Equation 7 to fine-tune. In practice, we find this procedure slightly improve the quality of the texture, especially in the areas that are not well-covered by the camera views as illustrated in Figure 4. However, since solving Equation 7 requires a differentiable renderer such as [Laine et al. 2020], involves more decision variables, and needs to be performed during every denoising step, this procedure can considerably increase the inference cost. In practice, we suggest using Algorithm 1 for fast preview and use Equation 7 to generate final results offline.

In addition, we realize that our method can be generalized into a framework to add constraints between multiple denoising process, which is not limited to multi-view consistency constraints. As an example, we show that our method can be used to achieve a similar effect as blended latent diffusion [Avrahami et al. 2023], but extended to 3D texture domain. To this end, we can assign two sets of cameras, but adopts two separate text prompts for each set, respectively. We then either use Equation 5 to blend the texture or

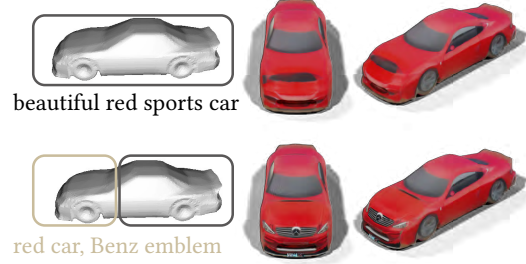


Figure 5: A car model with its generated texture from a global prompt (top) and different prompts for different local regions (bottom).

solve a joint optimization for two parts together using Equation 7. The results are illustrated in Figure 5, where the views from two types of cameras are blended seamlessly. We believe this approach can be further extended to multiple sets of prompts or other constraints between multiple diffusion process, which is left as future work.

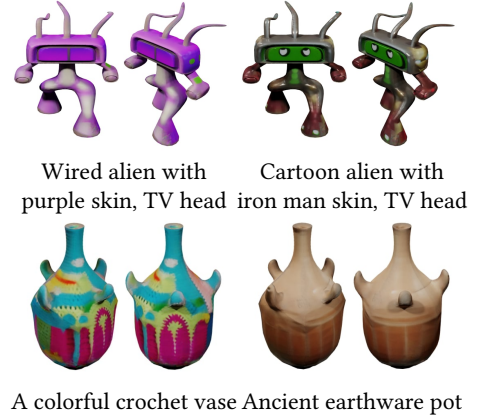


Figure 6: Generated textures from different prompts.

5 EVALUATION

All experiments are run on a machine with RTX 3090 GPU, an Intel Core i7-11600 CPU, and 16GB RAM. The total time for generating one texture is around 25 to 30 minutes. The pipeline utilizes the pre-trained LDM, Stable-Diffusion-2-Depth, which takes a depth map as the conditional input, generates a 2D latent code with $64 \times 64 \times 4$ resolution, and decodes it into an image with a resolution of $512 \times 512 \times 3$ in color space. To ensure a fair evaluation of our method, we adopt an identical configuration to render all mesh models in experiments. Initially, all meshes are normalized within a bounding box of unit length, and the UV are automatically mapped by XAtlas [jpcy [n. d.]]. Subsequently, we sample 8 fixed camera positions on a sphere with 1.5 radius, using a 45-degree field of view, 30-degree pitch, and evenly spaced yaw angles ranging from 0 to 315 degrees. Since illumination can disrupt the normal distribution during the reversed diffusion process, we opt not to apply lighting and shading during rendering. During the reversed diffusion process, we run 35 denoising steps. In each step, we optimize $\{\hat{z}_{0,t}^i\}$ using 20 iterations of AdamW optimizer with learning rate of 0.01. Afterward, we synthesis the final color texture from

all camera views. To accomplish this, we employ an optimization to match the color of texels with the corresponding colors in image views, using 500 iterations of SGD. The resolution of final texture is $1024 \times 1024 \times 3$. We show the results from our system in Figure 1. In Figure 6, our method works well with different text prompts on the same input mesh. We also display more generation results in the supplementary material.

We focus the comparison on TEXTure [Richardson et al. 2023], TexFusion [Cao et al. 2023], Meshy, and Fantasia3D [Chen et al. 2023a]. TEXTure performs the denoising process sequentially for each view and paints the texture in RGB space directly. The LDM they used is same with our backbone. TexFusion tightly couples the denoising process and latent-space fusion. Meshy is an industrial tool for texture generation. Fantasia3D is a SDS-based method, generating both models and textures. It also supports the synthesis of textures from a geometry input separately. Our evaluation is performed over 64 high-quality mesh models selected from 3D model datasets and multiple online resources and we use 182 pairs of meshes and text prompts for the experiments.

Qualitative Comparison. We displayed the texture generation results and the comparison of other four methods in Figure 10. In this experiment, we used the official implementation provided by TEXTure, Meshy and Fantasia3D. As the code for TexFusion is not released, we made a local implementation for this comparison. From the results, we found the high quality of our generated textures were consistent across all these models, while other texture painting methods had different problems with multi-view consistency. Additionally, the results from Fantasia3D sometimes exhibit over-exposure and over-saturation in color. Of the four techniques, we observe that the generation process in Fantasia3D exhibits issues with unnatural color generation. The TEXTure exhibits relatively worse result, exhibiting noisy texture patches. The results generated by Meshy have a high variation, giving stunning results on some models but low-quality textures on others. The results by TexFusion have the highest quality of the four while being consistently better than either TEXTure or Meshy. However, by detailed observation, our method achieves better quality and multi-view consistency. In addition to the comparison with our local implementation of TexFusion, we also reproduced several results with the same input meshes and text prompts as [Cao et al. 2023]. We supplemented the side-by-side comparison in Figure 8, where we adjusted our camera views and lighting conditions as much as possible for a fair comparison.

Quantitative Comparison. We further compared the quality of textures using the FID [Heusel et al. 2017] metric. This is a learned metric to measure the similarity between two datasets of images, in terms of visual quality and human preference. Since our method and TEXTure use SD2-depth as backbone, and Fantasia uses SD2, we propose to use the images generated by SD2-depth as the groundtruth dataset. Specifically, after generating our texture, we rendered the model under the given set of 8 camera views to generate our dataset. We then queried SD2-depth conditioned on the depth map from each camera view, along with the text prompt, to generate the groundtruth dataset. The resulting FID metrics for the four methods are summarized in Table 1. The FID scores reflect the same observations we made in our qualitative comparison. Note that

this comparison can be unfair for Meshy because we do not know their backbone LDM model. But since SD2-depth is trained using large-scale internet dataset, we assume the comparison is reasonable. Besides, we also measure the average time consumption of all methods during in the texture generation. The measurements are summarized in Table 2.

Ablation Study. Finally, we show that each and every component of our method is necessary. First, we highlighted the benefits of working with color space. In the experiment, we directly blended $\{\hat{z}_{0,t}^i\}$ into a latent texture $I_{z,0,t}$ and then rendered it under corresponding view to yield $\{z_{0,t}^i\}$, keeping other components intact. As shown in Figure 9 (left), this variant resulted in inconsistent and blurry textures. Next, we show that our method could only work with DDIM because it predicted a noiseless latent $\{\hat{z}_{0,t}^i\}$. For comparison, we combined our method with DDPM [Ho et al. 2020] and applied our color fusion on the noisy latent $\{z_t^i\}$, i.e., we decoded $\{z_t^i\}$ to $\{x_t^i\}$, blended it into $I_{x,0,t}$ and then optimized to yield $\{z_{0,t}^i\}$. As shown in Figure 9 (middle), this variant resulted in meaningless textures. This was due to the inconsistent noise component across views. In Equation 5, we employs an optimization to achieve the conversion from a color image to a latent code. A more direct approach is to utilize the VAE encoder of LDM built-in. However, frequent encoding of the image during generation can slightly decreases the quality of the final texture in some cases, despite significantly faster computation. In our quantitative experiment, we also measure the FID score and time consumption of our method with a VAE encoding, and the results are presented in Table 1 and Table 2. We illustrate a case that encounters encoding issues in Figure 9 (right).

Table 1: FID score of the five methods and our method with direct encoding(DE) under comparison.

TEXTure	Meshy	Fantasia3D	TexFusion	Ours	Ours(DE)
88.16	70.49	57.25	41.55	38.21	43.40

Table 2: The time consumption of the five methods and our method with direct encoding(DE) under comparison in minutes.

TEXTure	Meshy	Fantasia3D	TexFusion	Ours	Ours(DE)
5.5	N/A	24.1	4.2	27.3	4.8

6 CONCLUSION

We present TexPainter, a new approach to generate semantic textures for arbitrary, given 3D models, utilizing a pre-trained LDM image generator. Although this problem has been studied by prior works, we show that our method achieves better texture quality, via a new approach to enforce multi-view consistency. Our main idea is to utilize the predicted noiseless state in DDIM to perform multi-view fusion in the color space and then update the noiseless state using an optimization. Our approach further eliminates the assumption on sequential dependency between views. Through extensive experiments and comparisons, we confirm that our method delivers consistently high quality textures. We further highlight

the potential of our method as a general method for enforcing constraints between multiple diffusion process.

However, we still notice several shortcomings that worth further exploration. First, our method still uses a fixed set of camera views, while prior work [Chen et al. 2023c] proposed an algorithm to select a dynamic set of views for better covering of the model. We emphasize that it is non-trivial to combine a dynamic set of views with our method, because images from all the views must be present and fused together during each denoising step, making incremental view selection difficult. Further, our method inherits the limitations of prior works, i.e., the texture quality is limited by the pre-trained LDM and the sharp features cannot be generated well. Due to the lack of specific 3D directional constraints on the 2D LDM, it can sometimes erroneously generate multiple faces for a human model. Additionally, inadequate optimization and the weighted average operation potentially lead to blurry regions and smooth edges. The problem can be resolved in multiple ways in the future, including using a better pre-trained model, a texture up-sampling approach such as [Sajjadi et al. 2017], or finding an optimal method for texture synthesis while maintaining consistency in color space. Besides, utilizing a fine-tune diffusion model [Zhang et al. 2023] could enhance the overall quality of generated texture or the alignment with specific requirements. In addition, our method incurs higher computational cost than prior works [Cao et al. 2023] due to the repeated optimization. A lossless encoder would circumvent the time-intensive optimization in Equation 5, thereby significantly enhancing the performance of our method.

ACKNOWLEDGMENTS

This work has been supported by the Natural Science Foundation of Jiangsu Province Project award BK20232008, BK20211159, Jiangsu Key Research and Development Plan under Grant award BE2023023, the Joint Fund Project award 8091B042206 and the Fundamental Research Funds for the Central Universities. It is also partially supported by the Innovation and Technology Commission of the HKSAR Government under the InnoHK initiative.

REFERENCES

- Omri Avrahami, Ohad Fried, and Dani Lischinski. 2023. Blended latent diffusion. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–11.
- Fan Bao, Michael Schwarz, and Peter Wonka. 2013. Procedural facade variations from a single layout. *ACM Trans. Graph.* 32, 1, Article 8 (feb 2013), 13 pages. <https://doi.org/10.1145/2421636.2421644>
- 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 (ICCV)*. 8918–8928.
- Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. 2016. Generative and discriminative voxel modeling with convolutional neural networks. *arXiv preprint arXiv:1608.04236* (2016).
- Tianshi Cao, Karsten Kreis, Sanja Fidler, Nicholas Sharp, and KangXue Yin. 2023. TextFusion: Synthesizing 3D Textures with Text-Guided Image Diffusion Models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.
- Dave Zhenyu Chen, Yawar Siddiqui, Hsin-Ying Lee, Sergey Tulyakov, and Matthias Niesner. 2023c. Text2Tex: Text-driven Texture Synthesis via Diffusion Models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 18558–18568.
- Guoning Chen, Gregory Esch, Peter Wonka, Pascal Müller, and Eugene Zhang. 2008. Interactive procedural street modeling. In *ACM SIGGRAPH 2008 papers*. 1–10.
- Qimin Chen, Zhiqin Chen, Hang Zhou, and Hao Zhang. 2023b. ShaDDR: Interactive Example-Based Geometry and Texture Generation via 3D Shape Detailization and Differentiable Rendering. In *SIGGRAPH Asia 2023 Conference Papers* (<conf-loc>, <city>Sydney</city>, <state>NSW</state>, <country>Australia</country>, </conf-loc>) (SA '23). Association for Computing Machinery, New York, NY, USA, Article 58, 11 pages. <https://doi.org/10.1145/3610548.3618201>
- Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. 2023a. Fantasia3D: Disentangling Geometry and Appearance for High-quality Text-to-3D Content Creation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.
- Yongwei Chen, Rui Chen, Jiabao Lei, Yabin Zhang, and Kui Jia. 2022a. TANGO: Text-driven Photorealistic and Robust 3D Stylization via Lighting Decomposition. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Zhiqin Chen, Kangxue Yin, and Sanja Fidler. 2022b. AUV-Net: Learning Aligned UV Maps for Texture Transfer and Synthesis. In *The Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Junyu Dong, Jun Liu, Kang Yao, Mike Chantler, Lin Qi, Hui Yu, and Muwei Jian. 2020. Survey of procedural methods for two-dimensional texture generation. *Sensors* 20, 4 (2020), 1135.
- Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. 2022. GET3D: A Generative Model of High Quality 3D Textured Shapes Learned from Images. In *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 31841–31854.
- Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA). Red Hook, NY, USA, 6629–6640.
- Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Vancouver, BC, Canada). Curran Associates Inc., Red Hook, NY, USA, Article 574, 12 pages.
- Jonathan Ho and Tim Salimans. 2021. Classifier-Free Diffusion Guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*.
- jpcy. [n. d.]. XAtlas. <https://github.com/jpcy/xatlas>
- Animesh Karnewar, Andrea Vedaldi, David Novotny, and Niloy J. Mitra. 2023. HOLODIFFUSION: Training a 3D Diffusion Model Using 2D Images. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 18423–18433.
- Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. 2020. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–14.
- 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 *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 300–309.
- Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan LI, and Jun Zhu. 2022. DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps. In *Advances in Neural Information Processing Systems*, Vol. 35. 5775–5787.
- Tom Mertens, Jan Kautz, Jiawen Chen, Philippe Bekaert, and Frédo Durand. 2006. Texture Transfer Using Geometry Correlation. *Rendering Techniques* 273, 10.2312 (2006), 273–284.
- 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 *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 12663–12673.
- Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. 2022. Text2Mesh: Text-Driven Neural Stylization for Meshes. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 13482–13492.
- Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.
- Pascal Müller, Peter Wonka, Simon Haegler, Andreas Ulmer, and Luc Van Gool. 2006. Procedural modeling of buildings. In *ACM SIGGRAPH 2006 Papers*. 614–623.
- Norman Müller, Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulò, Peter Kotschieder, and Matthias Niesner. 2023. DiffRF: Rendering-Guided 3D Radiance Field Diffusion. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 4328–4338.
- Michael Oechsle, Lars Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger. 2019. Texture Fields: Learning Texture Representations in Function Space. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 4530–4539.
- Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. 2023. DreamFusion: Text-to-3D using 2D Diffusion. In *The Eleventh International Conference on Learning Representations*.
- Przemyslaw Prusinkiewicz, Mark Hammel, Jim Hanan, and Radomir Mech. 1996. L-systems: from the theory to visual models of plants. In *Proceedings of the 2nd CSIRO Symposium on Computational Challenges in Life Sciences*, Vol. 3. 1–32.
- 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* (Los Angeles, CA, USA) (SIGGRAPH '23). Association for Computing Machinery, New York, NY, USA, Article 54, 11 pages.

- Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 10674–10685.
- Mehdi SM Sajjadi, Bernhard Scholkopf, and Michael Hirsch. 2017. Enhancenet: Single image super-resolution through automated texture synthesis. In *Proceedings of the IEEE international conference on computer vision*. 4491–4500.
- Dominik Sibling, Darko Pavić, and Leif Kobbelt. 2010. Image synthesis for branching structures. In *Computer Graphics Forum*, Vol. 29. Wiley Online Library, 2135–2144.
- Yawar Siddiqui, Justus Thies, Fangchang Ma, Qi Shan, Matthias Niesner, and Angela Dai. 2022. Texturify: Generating Textures on 3D Shape Surfaces. In *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part III (Lecture Notes in Computer Science, Vol. 13663)*. Springer, 72–88. https://doi.org/10.1007/978-3-031-20062-5_5
- Ruben M Smelik, Tim Tutenel, Rafael Bidarra, and Bedrich Benes. 2014. A survey on procedural modelling for virtual worlds. In *Computer graphics forum*, Vol. 33. Wiley Online Library, 31–50.
- Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*. PMLR, 2256–2265.
- Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021a. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations*.
- Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2021b. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*.
- Ruoxi Sun, Jinyuan Jia, and Marc Jaeger. 2009. Intelligent tree modeling based on L-system. In *2009 IEEE 10th International Conference on Computer-Aided Industrial Design & Conceptual Design*. IEEE, 1096–1100.
- Jerry O Talton, Yu Lou, Steve Lesser, Jared Duke, Radomir Mech, and Vladlen Koltun. 2011. Metropolis procedural modeling. *ACM Trans. Graph.* 30, 2 (2011), 11–1.
- Shitao Tang, Fuyang Zhang, Jiacheng Chen, Peng Wang, and Yasutaka Furukawa. 2023. MVDiffusion: Enabling Holistic Multi-view Image Generation with Correspondence-Aware Diffusion. *arXiv* (2023).
- Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, Karsten Kreis, et al. 2022. LION: Latent Point Diffusion Models for 3D Shape Generation. *Advances in Neural Information Processing Systems* 35 (2022), 10021–10039.
- Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, and Thomas Wolf. 2022. Diffusers: State-of-the-art diffusion models.
- Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. 2023. ProlificDreamer: High-Fidelity and Diverse Text-to-3D Generation with Variational Score Distillation. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Li-Yi Wei, Sylvain Lefebvre, Vivek Kwatra, and Greg Turk. 2009. State of the art in example-based texture synthesis. *Eurographics 2009, State of the Art Report, EG-STAR* (2009), 93–117.
- Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. 2016. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *Advances in neural information processing systems* 29 (2016).
- Kai Yu, Jinlin Liu, Mengyang Feng, Miaomiao Cui, and Xuansong Xie. 2023b. Boosting3D: High-Fidelity Image-to-3D by Boosting 2D Diffusion Prior to 3D Prior with Progressive Learning. *arXiv:2311.13617 [cs.CV]*
- Rui Yu, Yue Dong, Pieter Peers, and Xin Tong. 2021. Learning texture generators for 3d shape collections from internet photo sets. In *British Machine Vision Conference*.
- Xin Yu, Peng Dai, Wenbo Li, Lan Ma, Zhengzhe Liu, and Xiaojuan Qi. 2023a. Texture Generation on 3D Meshes with Point-UV Diffusion. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. 4183–4193.
- Xianfang Zeng, Xin Chen, Zhongqi Qi, Wen Liu, Zibo Zhao, Zhibin Wang, Bin Fu, Yong Liu, and Gang Yu. 2023. Paint3D: Paint Anything 3D with Lighting-Less Texture Diffusion Models. *arXiv:2312.13913 [cs.CV]*
- Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. 2022. LION: Latent Point Diffusion Models for 3D Shape Generation. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- 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.
- Linqi Zhou, Yilun Du, and Jiajun Wu. 2021. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 5826–5835.
- Qi Zuo, Yafei Song, Jianfang Li, Lin Liu, and Liefeng Bo. 2023. DG3D: Generating High Quality 3D Textured Shapes by Learning to Discriminate Multi-Modal Diffusion-Renderings. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. 14529–14538. <https://doi.org/10.1109/ICCV51070.2023.01340>

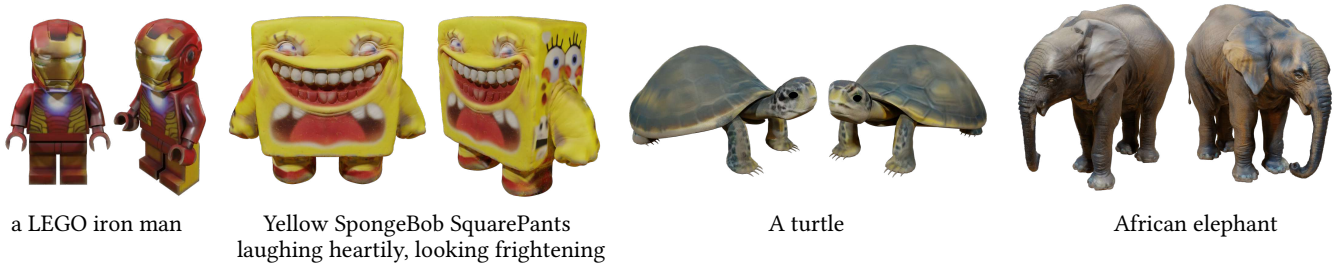


Figure 7: More results generated using our method.



Figure 8: Comparison between TexFusion and our method. We generate textures on the same meshes with the same prompts used in TexFusion (top) and render our results using similar views and lighting conditions (bottom).

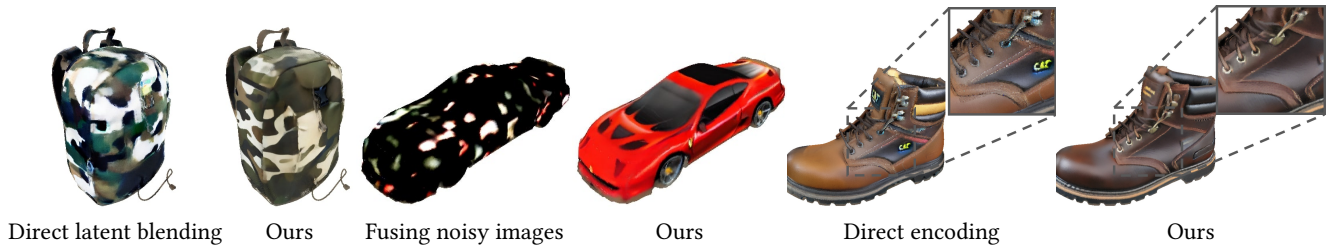


Figure 9: Ablation study of our proposed method. Left: Compared with our method that works in color space, directly blending the latents would result in inconsistent and blurry textures. Middle: If we blend the noisy images predicted using DDPM instead of using the noiseless images predicted using DDIM, the resulting textures could be meaningless. Right: The repeated direct encoding decreases the quality of the final texture.

TEXTure



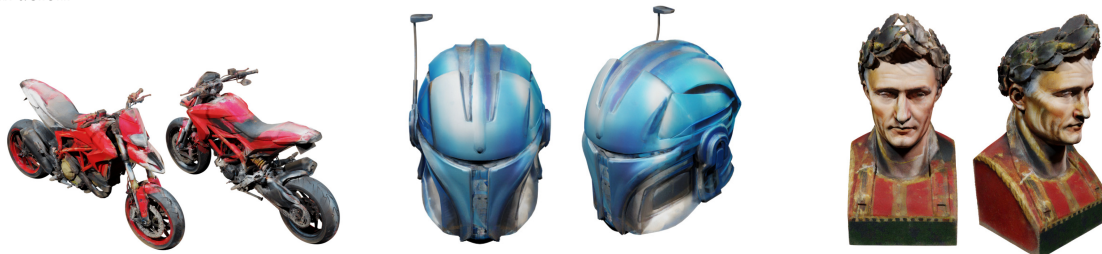
Meshy



Fantasia3D



TexFusion



Ours



motorbike, Ducati Hypermotard 939

a blue helmet, sci-fi movie

Julius Caesar, oil painting, full color

Figure 10: Comparison of textures generated by TEXTure, Meshy, Fantasia3D, our implementation of TexFusion, and our method.