TEXTure: Text-Guided Texturing of 3D Shapes

Anonymous Authors

https://texturepaper.github.io/TEXTurePaper/

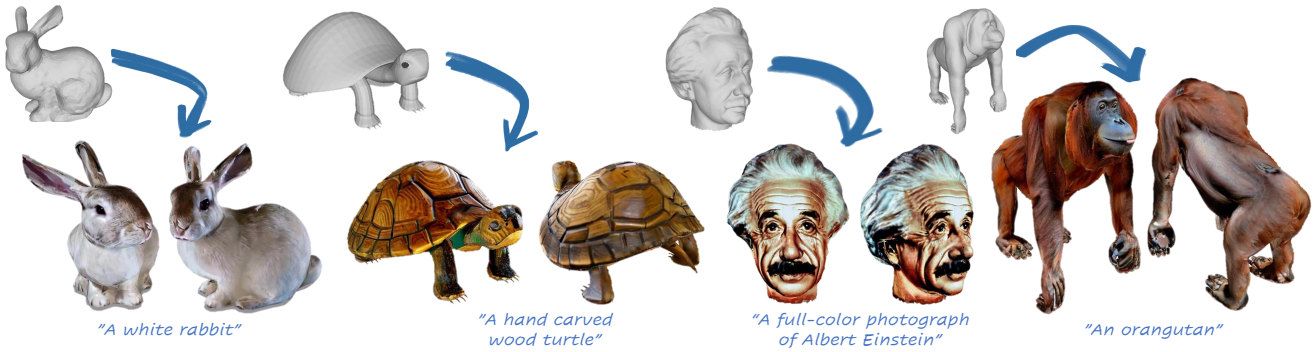

Figure 1. Texturing results. TEXTure takes an input mesh and a conditioning text prompt and paints the mesh with high-quality textures.

Abstract

In this paper, we present TEXTure, a novel method for text-guided generation, editing, and transfer of textures for 3D shapes. Leveraging a pretrained depth-to-image diffusion model, TEXTure applies an iterative scheme that paints a 3D model from different viewpoints. Yet, while depth-to-image models can create plausible textures from a single viewpoint, the stochastic nature of the generation process can cause many inconsistencies when texturing an entire 3D object. To tackle these problems, we dynamically define a trimap partitioning of the rendered image into three progression states, and present a novel elaborated diffusion sampling process that uses this trimap representation to generate seamless textures from different views. We then show that one can transfer the generated texture maps to new 3D geometries without requiring explicit surface-to-surface mapping, as well as extract semantic textures from a set of images without requiring any explicit reconstruction. Finally, we show that TEXTure can be used to not only generate new textures but also edit and refine existing textures using either a text prompt or user-provided scribbles. We demonstrate that our TEXTuring method excels at generating, transferring, and editing textures through extensive evaluation, and further close the gap between 2D image generation and 3D texturing. Code is available on: https://texturepaper.github.io/TEXTurePaper/

1. Introduction

The ability to paint pictures with words has long been a sign of a master storyteller, and with recent advancements in text-to-image models, this has become a reality for us all. Given a textual description, these new models are able to generate highly detailed imagery that captures the essence and intent of the input text. Despite the rapid progress in text-to-image generation, painting 3D objects remains a significant challenge as it requires to consider the specific shape of the surface being painted. Recent works have begun making significant progress in painting and texturing 3D objects by using language-image models as guidance [9, 28, 30, 51]. Yet, these methods still fall short in terms of quality compared to their 2D counterparts.

In this paper, we focus on texturing 3D objects, and present TEXTure, a technique that leverages diffusion models [38] to seamlessly paint a given 3D input mesh. Unlike previous texturing approaches [25, 28] that apply score distillation [33] to indirectly utilize Stable Diffusion [38] as a texturing prior, we opt to directly apply a full denoising process on rendered images using a depth-conditioned diffusion model [38].

At its core, our method iteratively renders the object from different viewpoints, applies a depth-based painting scheme, and projects it back to an atlas. We show that our approach can result in a significant boost in both running time and generation quality. However, applying this process naïvely would result in highly inconsistent texturing with noticeable seams due to the stochastic nature of the generation process (see Figure 2 (A)).

A B C D

Figure 2. Ablation of our different components: (A) is a naïve painting scheme. In (B) we introduce *"keep"* region. (C) is our improved scheme for *"generate"* regions, and (D) is our complete scheme with *"refine"* regions.

To alleviate these inconsistencies, we introduce a dynamic partitioning of the rendered view to a trimap of *"keep"*, *"refine"*, and *"generate"* regions, which is estimated before each diffusion process. The *"generate"* regions are areas in the rendered viewpoint that are viewed for the first time and need to be painted. A *"refine"* region is an area that was already painted in previous iterations, but is now seen from a better angle and should be repainted. Finally, *"keep"* regions are painted regions that should not be repainted from the current view. We then propose a modified diffusion process that takes into account our trimap partitioning. By freezing *"keep"* regions during the diffusion process we attain more consistent outputs, but the newly generated regions still lack global consistency (see Figure 2 (B)). To encourage better global consistency in the *"generate"* regions, we further propose to incorporate both depth-guided and mask-guided diffusion models into the sampling process (see Figure 2 (C)). Finally, for *"refine"* regions, we design a novel process that repaints these regions but takes into account their existing texture. Together these techniques allow the generation of highly-realistic results in mere minutes (see Figure 2 (D) and Figure 1 for results).

Next, we show that our method can be used not only to texture meshes guided by a text prompt, but also based on an existing textures from some other colored mesh or even from a small set of images. Our method requires no surface-to-surface mapping or any intermediate reconstruction step. Instead, we propose to learn semantic tokens that represent a specific texture by building on Textual Inversion [15] and DreamBooth [39], while extending them to depth-conditioned models and introducing learned viewpoint tokens. We show that we can successfully capture the essence of a texture even from a few unaligned images and use it to paint a 3D mesh based on its semantic texture.

Finally, in the spirit of diffusion-based image editing [20–22, 48], we show that one can further refine and edit textures. We propose two editing techniques. First, we present a text-only refinement where an existing texture map is modified using a guiding prompt to better match the semantics of the new text. Second, we illustrate how users can directly apply an edit on a texture map, where we refine the texture to fuse the user-applied edits into the 3D shape.

We evaluate TEXTure and show its effectiveness for texture generation, transfer, and editing. We demonstrate that TEXTure offers a significant speedup compared to previous approaches, and more importantly, offers significantly higher-quality generated textures.

2. Related Work

Text-to-Image Diffusion Models. The past year has seen the development of multiple large diffusion models [31, 36, 38, 40] capable of producing impressive images with pristine details guided by an input text prompt. The widely popular Stable Diffusion [38], is trained on a rich text-image dataset [43] and is conditioned on CLIP's [35] frozen text encoder. Beyond simple text-conditioning, Stable Diffusion has multiple extensions that allow conditioning its denoising network on additional input modalities such as a depth map or an inpainting mask. Given a guiding prompt and an estimated depth image [37], the depth-conditioning model is tasked with generating images that follow the same depth values while being semantically faithful with respect to the text. Similarly, the inpainting model completes missing image regions, given a masked image.

Although current text-to-image models generate high-quality results when conditioned on a text prompt or depth map, editing an existing image or injecting objects specified by a few exemplar images remains challenging [2, 47]. To introduce a user-specific concept to a pre-trained text-to-image model, [15] introduce Textual Inversion to map a few exemplar images into a learned pseudo-tokens in the embedding space of the frozen text-to-image model. DreamBooth [39] further fine-tune the entire diffusion model on the set of input images to achieve more faithful compositions. The learned token or fine-tuned model can then be used to generate novel images using the custom token in new user-specified text prompts.

Texture and Content Transfer. Early works [11, 12, 18, 54] focus on 2D texture synthesis through probabilistic models, while more recent works [13, 45, 50, 53] take a data-driven approach to generate textures using deep neural networks. Generating textures over 3D surfaces is a more challenging problem, as it requires attention to both colors and geometry. For geometric texture synthesis, [17] applies a similar statistical method to [18] while [6] extended non-parametric sampling proposed by [12] to 3D meshes. [5] introduces a metric learning approach to transfer details from a source to a target shape, while [19] use an internal learning technique to transfer geometric texture. For 3D color texture synthesis, given an exemplar colored mesh, [8, 26, 27] use the relation between geometric features and color values to synthesize new textures on target shapes.

3D Shape and Texture Generation. Generating shapes and textures in 3D has recently gained significant interest. Text2Mesh [30], Tango [9], and CLIP-Mesh [23] use CLIP-space similarities as an optimization objective to generate novel shapes and textures. CLIP-Mesh deforms an initial sphere with UV texture parameterization. Tango optimizes per-vertex colors and focuses on generating novel textures. Text2Mesh optimizes per-vertex color attributes, while allowing small geometric displacements. Get3D [16] is trained to generate shape and texture through a DMTet [46] mesh extractor and 2D adversarial losses.

Recently, DreamFusion [33] introduced the use of pretrained image diffusion models for generating 3D NeRF models conditioned on a text prompt. The key component in DreamFusion is the *Score-Distillation* loss which enables the use of a pretrained 2D diffusion model as a critique for optimizing the 3D NeRF scene. Recently, Latent-NeRF [28] showed how the same Score-Distillation loss can be used in Stable Diffusion's latent space to generate latent 3D NeRF models. In the context for texture generation, [28] present *Latent-Paint*, a texture generation technique, where latent texture maps are painted using Score-Distillation and are then decoded to RGB for the final colorization output. Similarly, [25] uses score-distillation to texture and refine a coarse initial shape. Both methods suffer from relatively slow convergence and less defined textures compared to our proposed approach.

3. Method

We first lay the foundation for our text-guided mesh texturing scheme, illustrated in Figure 3. Our TEXTure scheme performs an incremental texturing of a given 3D mesh, where at each iteration we paint the currently visible regions of the mesh as seen from a single viewpoint. To encourage both local and global consistency, we segment the mesh into a trimap of *"keep"*, *"refine"*, *"generate"* regions. A modified depth-to-image diffusion process is presented to incorporate this information into the denoising steps.

We then propose two extensions of TEXTure. First, we present a texture transfer scheme (Section 3.2) that transfers the texture of a given mesh to a new mesh, by learning a custom concept that represents the given texture. Finally, we present a texture editing technique that allows users to edit a given texture map, either through a guiding text prompt or a user-provided scribble (Section 3.3).

3.1. Text-Guided Texture Synthesis

Our texture generation method relies on a pretrained depth-to-image diffusion model $\mathcal{M}_{depth}$ and a pretrained inpainting diffusion model $\mathcal{M}_{paint}$, both based on Stable Diffusion [38] and with a shared latent space. During the generation process, the texture is represented as an atlas through a UV mapping that is calculated using XAtlas [52].

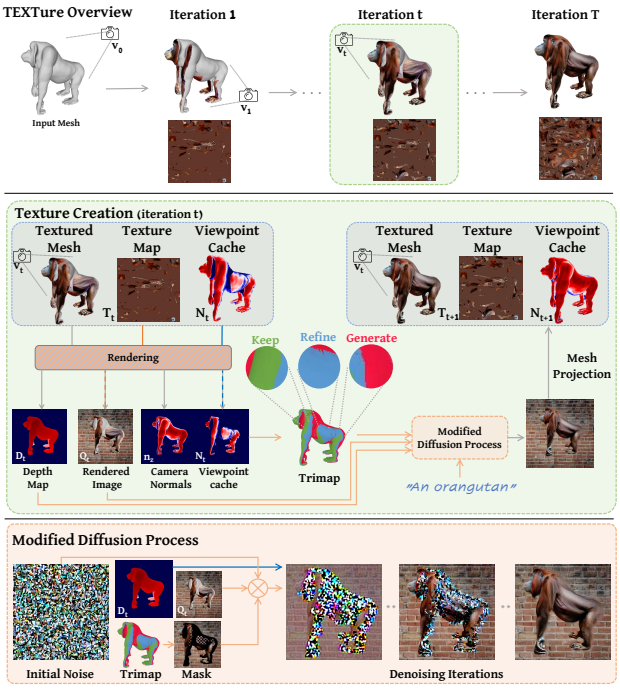

Figure 3. Our schematic texturing process. A mesh is iteratively painted from different viewpoints. In each painting iteration, we render the mesh alongside its depth and normal maps. We then calculate a trimap partitioning of the image to three distinct areas based on the camera normals and a viewpoint cache representing previously viewed angles. These inputs are then fed into a modified diffusion process alongside a given text prompt which generates an updated image. This image is then projected back to the texture map for the next iteration.

We start from an arbitrary initial viewpoint $v_0 = (r = 1.25, \phi_0 = 0, \theta = 60)$ where r is the radius of the camera, ϕ is the azimuth camera angle, and θ is the camera elevation. We then use $\mathcal{M}_{depth}$ to generate an initial colored image I_0 of the mesh as viewed from v_0, conditioned on the rendered depth map $\mathcal{D}_0$. The generated image I_0 is then projected back to the texture atlas $\mathcal{T}_0$ to color the shape's visible parts from v_0. Following this initialization step, we begin a process of incremental colorization, illustrated in Figure 3, where we iterate through a fixed set of viewpoints. For each viewpoint, we then render the mesh using a renderer $\mathcal{R}$ [14] to obtain $\mathcal{D}_t$ and Q_t, where Q_t is the rendering of the mesh as seen from the viewpoint v_t that considers all previous colorization steps. Finally, we generate the next image I_t and project I_t back to the updated texture atlas $\mathcal{T}_t$ while taking into account Q_t.

Once a single view has been painted, the generation task becomes more challenging due to the need for local and global consistency along the generated texture. Below we consider a single iteration t of our incremental painting process and elaborate on our proposed techniques to handle these challenges.

Trimap Creation. Given a viewpoint v_t, we first apply a partitioning of the rendered image into three regions: *"keep"*, *"refine"*, and *"generate"*. The *"generate"* regions are rendered areas that are viewed for the first time and need to be painted to match the previously painted regions. The distinction between *"keep"* and *"refine"* regions is slightly more nuanced and is based on the fact that coloring a mesh from an oblique angle can result in high distortion. This is because the cross-section of a triangle with the screen is low, resulting in a low-resolution update to the mesh texture image $\mathcal{T}_t$. Specifically, we measure the triangle's cross-section as the z component of the face normal n_z in the camera's coordinate system.

Ideally, if the current view provides a better colorization angle for some of the previously-painted regions, we would like to *"refine"* their existing texture. Otherwise, we should *"keep"* the original texture and avoid modifying it to ensure consistency with previous views. To keep track of seen regions and the cross-section at which they were previously colored from, we use an additional meta-texture map $\mathcal{N}$ that is updated at every iteration. This additional map can be efficiently rendered together with the texture map at each iteration and is used to define the current trimap partitioning.

Masked Generation. As the depth-to-image diffusion process was trained to generate an entire image, we must modify the sampling process to *"keep"* part of the image fixed. Following Blended Diffusion [2, 3], in each denoising step we explicitly inject a noised versions of Q_t, *i.e.* z_{Q_t}, at the *"keep"* regions into the diffusion sampling process, such that these areas are seamlessly blended into the final generated result. Specifically, the latent at the current sampling timestep i is computed as

$$z_i \leftarrow z_i \odot m_{blended} + z_{Q_t} \odot (1 - m_{blended}) \quad (1)$$

where the mask $m_{blended}$ is defined in Equation 2. That is, for *"keep"* regions, we simply set z_i fixed according to their original values.

Consistent Texture Generation. Injecting *"keep"* regions into the diffusion process results in better blending with *"generate"* regions. Still, when moving away from the *"keep"* boundary and deeper into the *"generate"* regions, the generated output is mostly governed by the sampled noise and is not consistent with previously painted regions. We first opt to use the same sampled noise from each viewpoint, this sometimes improves consistency, but is still very sensitive to the change in viewpoint. We observe that applying an inpainting diffusion model $\mathcal{M}_{paint}$ that was directly trained to complete masked regions, results in more consistent generations. However, this in turn deviates from the conditioning depth $\mathcal{D}_t$ and may generate new geometries. To benefit from the advantages of both models we introduce an interleaved process where we alternate between the two models during the initial sampling steps. Specifically, during sampling, the next noised latent z_{i-1} is computed as:

$$z_{i-1} = \begin{cases} \mathcal{M}_{depth}(z_i, \mathcal{D}_t) & 0 \le i < 10 \\ \mathcal{M}_{paint}(z_i, \text{"generate"}) & 10 \le i < 20 \\ \mathcal{M}_{depth}(z_i, \mathcal{D}_t) & 20 \le i < 50 \end{cases}$$

When applying $\mathcal{M}_{depth}$, the noised latent is guided by the current depth $\mathcal{D}_t$ while when applying $\mathcal{M}_{paint}$, the sampling process is tasked with completing the *"generate"* regions in a globally-consistent manner.

Refining Regions. To handle *"refine"* regions we use another novel modification to the diffusion process that generates new textures while taking into account their previous values. Our key observation is that by using an alternating checkerboard-like mask in the first steps of the sampling process, we can guide the noise towards values that locally align with previous completions.

The granularity of this process can be controlled by changing the resolution of the checkerboard mask and the number of constrained steps. In practice, we apply the mask for the first 25 sampling steps. Namely, the masked $m_{blended}$ applied in Equation (1) is set as,

$$m_{blended} = \begin{cases} 0 & \text{"keep"} \\ \text{checkerboard} & \text{"refine"} \wedge i \le 25 \\ 1 & \text{"refine"} \wedge i > 25 \\ 1 & \text{"generate"} \end{cases} \quad (2)$$

where a value of 1 indicates that this region should be painted and kept otherwise. Our blending mask is visualized in Figure 3.

Texture Projection. To project I_t back to the texture atlas $\mathcal{T}_t$, we apply gradient-based optimization for $\mathcal{L}_t$ over the values of $\mathcal{T}_t$ when rendered through the differential renderer $\mathcal{R}$. That is,

$$\nabla_{\mathcal{T}_t} \mathcal{L}_t = [(\mathcal{R}(mesh, \mathcal{T}_t, v_t) - I_t) \odot m_s] \frac{\partial \mathcal{R} \odot m_s}{\partial \mathcal{T}_t}$$

To achieve smoother texture seams of the projections from different views, a soft mask m_s is applied at the boundaries of the *"refine"* and *"generate"* region:

$$m_s = m_h * g \qquad m_h = \begin{cases} 0 & \text{"keep"} \\ 1 & \text{"refine"} \cup \text{"generate"} \end{cases}$$

where g is a 2D Gaussian blur kernel.

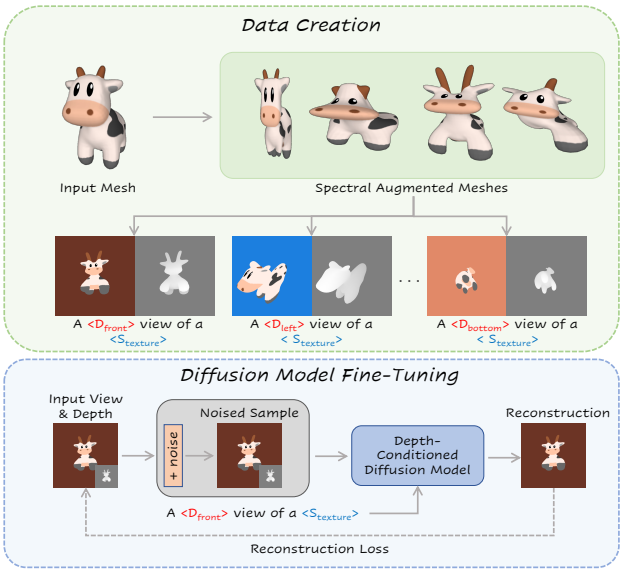

Figure 4. Fine-Tuning for texture transfer. (top) Given an input mesh, spectral augmentations are applied to generate a variety of textured geometries. The geometries are then rendered from random viewpoints, where each image is coupled with a sentence, based on the viewpoint. (bottom) a depth-conditioned diffusion model is then fine-tuned with a set of learnable tokens, a fixed $\langle \mathcal{S}_{texture} \rangle$ token representing the object, and an additional viewpoint token, $\langle \mathcal{D}_v \rangle$. The tuned model is used to paint new objects.

Additional Details. Our texture is represented as a 1024×1024 atlas, where the rendering resolution is 1200×1200. For the diffusion process, we segment the inner region, resize it to 512×512 and mat it onto a realistic background. All shapes are rendered with 8 viewpoints around the object, and two additional top/bottom views. We show that viewpoint order can also affect the end results.

3.2. Texture Transfer

Having successfully generated a new texture on a given 3D mesh, we now turn to describe how to *transfer* a given texture to a new, untextured target mesh. We show how to capture textures from either a painted mesh, or from a small set of input images. Our texture transfer approach builds on previous work on concept learning over diffusion models [15, 39], by fine-tuning a pretrained diffusion model and learning a pseudo-token representing the generated texture. The fine-tuned model is then used for texturing a new geometry. To improve the generalization of the fine-tuned model to new geometries, we further propose a novel *spectral augmentation* technique, described next. We then discuss our concept learning scheme from meshes or images.

Spectral Augmentations. Since we are interested in learning a token representing the input **texture** and not the original input geometry itself, we should ideally learn a common token over a range of geometries containing the input texture. Doing so disentangles the texture from its specific geometry and improves the generalization of the fine-tuned diffusion model. Inspired by the concept of surface caricaturization [44], we propose a novel *spectral augmentation* technique. In our case, we propose random low-frequency geometric deformations to the textured source mesh, regularized by the mesh Laplacian's spectrum [29].

Modulating random deformations over the spectral eigenbasis results in smooth deformations that keep the integrity of the input shape. Empirically, we choose to apply random inflations or deflations to the mesh, with a magnitude proportional to a randomly selected eigenfunction. We provide examples of such augmentations in Figure 4, and additional details in the supplementary materials.

Texture Learning. Applying our spectral augmentation technique, we generate a large set of images with corresponding depth maps of the input shape. We render the images from several viewpoints (left, right, overhead, bottom, front, and back) and paste the rendered object onto a randomly colored background (see Figure 4).

Given the set of rendered images, we follow [15] and optimize an embedding vector representing our texture using prompts of the form "a $\langle D_v \rangle$ photo of a $\langle \mathcal{S}_{texture} \rangle$" where $\langle D_v \rangle$ is a learned token representing the view direction of the rendered image and $\langle \mathcal{S}_{texture} \rangle$ is the token representing our texture. Observe, that we have six learned directional tokens D_v, shared within images from the same view, and a single token $\mathcal{S}_{texture}$ representing the texture, shared across all images. Additionally, to better capture the input texture we fine-tune the diffusion model itself as well, as done in [39]. Our texture learning scheme is illustrated in Figure 4. After training, we use TEXTure (Section 3.1), to color the target shape, swapping the original Stable Diffusion model with our fine-tuned model.

Texture from Images. Next, we explore the more challenging task of texture generation based on a small set of sample images. While we cannot expect the same quality given only a few images, we can still potentially learn concepts that represent different textures. Unlike standard textual inversion techniques [15, 39] our learned concepts represent mostly texture and not structure as they are trained on a depth-conditioned model. This potentially makes them more suitable for texturing other 3D shapes.

For this task, we segment the prominent object from the image using a pretrained saliency network [34], apply standard scale and crop augmentations, and paste the result onto a randomly-colored background. Our results show that one can successfully learn semantic concepts from images and apply them to 3D shapes without any explicit reconstruction stage in between. We believe this creates new opportunities for creating captivating textures inspired by real objects.

"A turtle" "A hand carved wood elephant" "A wooden klein bottle"

Figure 5. Texturing results. Our method generates a high-quality texture for a collection of prompts and geometries.

3.3. Texture-Editing

We show that our trimap-based TEXTuring can be used to easily extend 2D editing techniques to a full mesh. For text-based editing, we wish to alter an existing texture map guided by a textual prompt. To this end, we define the entire texture map as a *"refine"* region and apply our TEXTuring process to modify the texture to align with the new text prompt. We additionally provide scribble-based editing where a user can directly edit a given texture map (*e.g.* to define a new color scheme over a desired region). To allow this, we simply define the altered regions as *"refine"* regions during the TEXTuring process and *"keep"* the remaining texture fixed.

4. Experiments

We now turn to validate the robustness and effectiveness on our proposed method through a set of experiments.

4.1. Text-Guided Texturing

Qualitative Results. We first demonstrate results achieved with TEXTure across several geometries and driving text prompts. Figure 5, Figure 1 and Figure 11 show highly-detailed realistic textures that are conditioned on a single text prompt. Observe, for example, how the generated textures nicely align with the geometry of the turtle and elephant shapes. Moreover, the generated textures are consistent both on a local scale (e.g., along the shell of the turtle) and a global scale (e.g., the shell is consistent across different views). Furthermore, TEXTure can successfully generate textures of an individual (e.g., of Albert Einstein in Figure 1). Observe how the generated texture captures fine details of Albert Einstein's face. Finally, TEXTure can successfully handle challenging geometric shapes, including the non-orientable Klein bottle, shown in Figure 5.

Qualitative Comparisons. In Figure 6 we compare our method to several state-of-the-art methods for 3D texturing. First, we observe that Clip-Mesh [23] and Text2Mesh [30] struggle in achieving globally-consistent results due to their heavy reliance CLIP-based guidance. See the "90's boombox" example at the top of Figure 6 where speakers are placed sporadically in competing methods. For Latent-Paint [28], the results are more plausible but still lack in terms of quality. For example, Latent-Paint often struggles in achieving visibly sharp textures such as those shown on "'a desktop iMac'. We attribute this shortcoming due to its reliance on score distillation [33], which tends to omit high-frequency details.

Method	Overall Quality (↑)	Text Fidelity (↑)	Runtime (minutes) (↓)
Text2Mesh	2.57	3.62	32 (6.4×)
Latent-Paint	2.95	4.01	46 (9.2×)
TEXTure	**3.93**	**4.44**	**5**

Table 1. User study results conducted with 30 respondents. We ask respondents to rate the results on a scale of 1 to 5 with respect to the overall quality of the results and the level at which the result reflects the text prompt. Results are averaged across all responses and text prompts.

The last row in Figure 6, depicting a statue of Napoleon Bonaparte, showcases our method's ability to produce fine details compared to alternative methods that struggle to produce matching quality. Notably, our results were achieved significantly faster than the alternative methods. Generating a single texture with TEXTure takes approximately 5 minutes compared to 19 through 45 minutes for alternative

"A 90's boombox"

"A desktop iMac"

"A photo of Napoleon Bonaparte"

Clip-Mesh Text2Mesh Latent-Paint Ours

Figure 6. Visual comparison of text-guided texture generation. For each input prompt, we show results for a single viewpoint. Best viewed zoomed in. Meshes obtained from ModelNet40 [49].

methods (See Table 1). We provide additional qualitative results in 11 and in the supplementary materials.

User Study. Finally, we conduct a user study to analyze the fidelity and overall quality of the generated textures. We select 10 text prompts and corresponding 3D meshes and texture the meshes using TEXTure and two baselines: Text2Mesh [30] and Latent-Paint [28]. For each prompt and method, we ask each respondent to evaluate the result with respect to two aspects: (1) its overall quality and (2) the level at which it reflects the text prompt, on a scale of 1 to 5. Results are presented in Table 1 where we show the average results across all prompts for each method. As can be seen, TEXTure outperforms both baselines in terms of both overall quality and text fidelity by a significant margin. Importantly, our method's improved quality and fidelity are attained with a significant decrease in runtime. Specifically, TEXTure achieves a decrease of $6.4\times$ in running time compared to Text2Mesh and a decrease of $9.2\times$ relative to Latent-Paint.

In addition to the above evaluation setting, we ask respondents to rank the methods relative to each other. Specifically, for each of the 10 prompts, we show the results of all methods side-by-side (in a random order) and ask respondents to rank the results. In Table 2 we present the average rank of each method, averaged across the 10 prompts and across all responses. Note that a lower rank is preferable in this setting. As can be seen, TEXTure has a significantly better average rank relative to the two baselines, as desired. This further demonstrates the effectiveness of TEXTure in generating high-quality, semantically-accurate textures.

Ablation Study. An ablation validating the different components of our TEXTure scheme is shown in Figure 7. One can see that each component is needed for achieving high-quality generations and improving our method's robustness to the sensitivity of the generation process. Specifically, without differentiating between *"keep"* and *"generate"* regions the generated textures are unsatisfactory. For both the teddy bear and the sports car one can clearly see that the texture presented in A fails to achieve local and global consistency, with inconsistent patches visible across the texture. In contrast, thanks to our blending technique for *"keep"* regions, B achieves local consistency between views. Still, observe that the texture of the teddy bear legs in the top example, do not match the texture of its back. By incorporating our improved *"generate"* method, we achieve more consistent results across the entire shape, as shown in C. These results tend to have smeared regions as can be observed in the fur of the teddy bear, or the text written across the hood of the car. We attribute this to the fact that some regions are painted from oblique angles at early viewpoints, which are not ideal for texturing, and are not refined. By identifying *"refine"* regions and applying our full TEXTure scheme, we are able to effectively address these problems and produce sharper textures, see D.

Figure 7. Ablation of our different stages: (A) is a naïve painting scheme that paints the entire viewpoint. (B) takes into account *"keep"* region. (C) is our inpainting-based scheme for *"generate"* regions, and (D) is our complete scheme with *"refine"* regions. Car model obtained from [49], Teddy Bear model obtained from [32].

	Text2Mesh	Latent-Paint	**TEXTure**
Average Rank (↓)	2.44	2.24	**1.32**

Table 2. Additional user study results. Each respondent is asked to rank the results of the different methods with respect to overall quality.

Figure 8. Token-based Texture Transfer. The learned token ⟨$\mathcal{S}_{texture}$⟩ is applied to different geometries. Einstein's head is shown with both exact and "style" prompts, demonstrating the effect on the transfer's semantics. Both Spot and Ogre are taken from Keenan's Repository [10].

4.2. Texture Capturing

We next validate our proposed texture capture and transfer technique that can be applied over both 3D meshes and images.

Texture From Mesh As mentioned in Section 3.2 we are able to capture a texture of a given mesh by combining concept learning techniques [15, 39] with view-specific directional tokens. We can then use the learned token with TEXTure to color new meshes accordingly. Figure 8 presents texture transfer results from two existing meshes onto various target geometries. While the training of the new texture token is performed using prompts of the form "A $\langle D_v \rangle$ photo of $\langle \mathcal{S}_{texture} \rangle$", we can generate new textures by forming different texture prompts when transferring the learned texture. Specifically, the "Exact" results in Figure 8 were generated using the specific text prompt used for fine-tuning, and the rest of the outputs were generated "in the style of $\langle \mathcal{S}_{texture} \rangle$". Observe how a single eye is placed on Einstein's face when the exact prompt is used, while two eyes are generated when only using a prompt "a $\langle D_v \rangle$ photo of Einstein that looks like $\langle \mathcal{S}_{texture} \rangle$". A key component of the texture-capturing technique is our novel spectral augmentations scheme. We refer the reader to the supplementary materials for an ablation study over this component.

Texture From Image In practice, it is often more practical to learn a texture for a set of images rather than from a 3D mesh. As such, in Figure 9, we demonstrate the results of our transferring scheme where the texture is captured from a collection of images depicting the source object. One can see that even when given only several images, our method can successfully texture different shapes with a semantically similar texture. We find these results exciting, as it means one can easily paint different 3D shapes using textures derived from real-world data, even when the texture itself is only implicitly represented in the fine-tuned model and concept tokens.

4.3. Editing

Finally, in Figure 15 we show the editing results of existing textures. The top row of Figure 15 shows scribble-based results where a user manually edits the texture atlas image. Then we *"refine"* the texture atlas to seamlessly blend the edited region into the final texture. Observe how the manually annotated white spot on the bunny on the right turns into a realistic-looking patch of white fur.

The bottom of Figure 15 shows text-based editing results, where an existing texture is *"refined"* according to a new text prompt. The bottom right example illustrates a texture generated on a similar geometry with the same prompt **from scratch**. Observe that the texture generated from scratch significantly differs from the original texture. In contrast, when applying editing, the textures are able to remain semantically close to the input texture while generating novel details to match the target text.

Images Set Textured Meshes

Figure 9. Token-based Texture Transfer from images. New meshes are presented only once. All meshes are textured with the **exact** prompt. The teddy bear is also presented with the "..in the style of" prompt, one can see that this results in a semantic mix between the image set and a teddy bear.

Depth Inconsistencies Viewpoint Inconsistencies

Figure 10. Limitations. Left: input mesh, where the diffusion output was inconsistent with the geometry of the model face. These inconsistencies are clearly visible from other views and affect the painting process. Right: two different viewpoints of "a goldfish" where the diffusion process added inconsistent eyes, as the previously painted eyes were not visible.

5. Discussion, Limitations and Conclusions

This paper presents TEXTure, a novel method for text-guided generation, transfer, and editing of textures for 3D shapes. There have been many models for generating and editing high-quality images using diffusion models. However, leveraging these image-based models for generating seamless textures on 3D models is a challenging task, in particular for non-stochastic and non-stationary textures. Our work addresses this challenge by introducing an iterative painting scheme that leverages a pretrained depth-to-image diffusion model. Instead of using a computationally demanding score distillation approach, we propose a modified image-to-image diffusion process that is applied from a small set of viewpoints. This results in a fast process capable of generating high-quality textures in mere minutes. With all its benefits, there are still some limitations to our proposed scheme, which we discuss next.

While our painting technique is designed to be spatially coherent, it may sometimes result in inconsistencies on a global scale, caused by occluded information from other views. See Figure 10, where different looking eyes are added from different viewpoints. Another caveat is viewpoint selection. We use eight fixed viewpoints around the object, which may not fully cover adversarial geometries. This issue can possibly be solved by finding a dynamic set of viewpoints that maximize the coverage of the given

mesh. Furthermore, the depth-guided model sometimes deviates from the input depth, and may generate images that are not consistent with the geometry (See Figure 10 left). This in turn may result in conflicting projections to the mesh, that cannot be fixed in later painting iterations.

With that being said, we believe that TEXTure takes an important step toward revolutionizing the field of graphic design and further opens new possibilities for 3D artists, game developers, and modelers who can use these tools to generate high-quality textures in a fraction of the time of existing techniques. Additionally, our trimap partitioning formulation provides a practical and useful framework that we hope will be utilized and *"refined"* in future studies.

Mesh Credits

Meshes throughout this paper are taken from [1, 4, 7, 10, 24, 30, 32, 41, 42, 49].

References

[1] Arafurisan. Pikachu 3d sculpt free 3d model. https://www.cgtrader.com/free-3d-models/character/child/pikachu-3d-model-free, 2022. Model ID: 3593494. 9

[2] Omri Avrahami, Ohad Fried, and Dani Lischinski. Blended latent diffusion. *arXiv preprint arXiv:2206.02779*, 2022. 2, 4

[3] Omri Avrahami, Ohad Fried, and Dani Lischinski. Blended latent diffusion. *arXiv preprint arXiv:2206.02779*, 2022. 4

[4] behnam.aftab. Free 3d turtle model. https://3dexport.com/free-3dmodel-free-3d-turtle-model-355284.htm, 2021. Item ID: 355284. 9

[5] Sema Berkiten, Maciej Halber, Justin Solomon, Chongyang Ma, Hao Li, and Szymon Rusinkiewicz. Learning detail transfer based on geometric features. In *Computer Graphics Forum*, volume 36, pages 361–373. Wiley Online Library, 2017. 2

[6] Toby P Breckon and Robert B Fisher. A hierarchical extension to 3d non-parametric surface relief completion. *Pattern Recognition*, 45(1):172–185, 2012. 2

[7] RAMIRO CASTRO. Elephant natural history museum. https://www.cgtrader.com/free-3d-print-models/miniatures/other/elephant-natural-history-museum-1, 2020. Model ID: 2773650. 9

[8] Xiaobai Chen, Tom Funkhouser, Dan B Goldman, and Eli Shechtman. Non-parametric texture transfer using meshmatch. *Technical Report Technical Report 2012-2*, 2012. 2

[9] Yongwei Chen, Rui Chen, Jiabao Lei, Yabin Zhang, and Kui Jia. Tango: Text-driven photorealistic and robust 3d stylization via lighting decomposition. *arXiv preprint arXiv:2210.11277*, 2022. 1, 3

[10] Keenan Crane. Keenan's 3d model repository. https://www.cs.cmu.edu/ kmcrane/Projects/ModelRepository/, 2022. 7, 9

[11] Jeremy S De Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 361–368, 1997. 2

[12] Alexei A Efros and Thomas K Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1033–1038. IEEE, 1999. 2

[13] Anna Frühstück, Ibraheem Alhashim, and Peter Wonka. Tilegan: Synthesis of large-scale non-homogeneous textures. *ACM Trans. Graph.*, 38:58:1–58:11, 2019. 2

[14] Clement Fuji Tsang, Maria Shugrina, Jean Francois Lafleche, Towaki Takikawa, Jiehan Wang, Charles Loop, Wenzheng Chen, Krishna Murthy Jatavallabhula, Edward Smith, Artem Rozantsev, Or Perel, Tianchang Shen, Jun Gao, Sanja Fidler, Gavriel State, Jason Gorski, Tommy Xiang, Jianing Li, Michael Li, and Rev Lebaredian. Kaolin: A pytorch library for accelerating 3d deep learning research. https://github.com/NVIDIAGameWorks/kaolin, 2022. 3

[15] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022. 2, 5, 8

[16] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *arXiv preprint arXiv:2209.11163*, 2022. 3

[17] Aleksey Golovinskiy, Wojciech Matusik, Hanspeter Pfister, Szymon Rusinkiewicz, and Thomas Funkhouser. A statistical model for synthesis of detailed facial geometry. *ACM Transactions on Graphics (TOG)*, 25(3):1025–1034, 2006. 2

[18] David J Heeger and James R Bergen. Pyramid-based texture analysis/synthesis. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 229–238, 1995. 2

[19] Amir Hertz, Rana Hanocka, Raja Giryes, and Daniel Cohen-Or. Deep geometric texture synthesis. *ACM Trans. Graph.*, 39(4), 2020. 2

[20] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022. 2

[21] Dong Huk Park, Grace Luo, Clayton Toste, Samaneh Azadi, Xihui Liu, Maka Karalashvili, Anna Rohrbach, and Trevor Darrell. Shape-guided diffusion with inside-outside attention. *arXiv e-prints*, pages arXiv–2212, 2022. 2

[22] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. *arXiv preprint arXiv:2210.09276*, 2022. 2

[23] Nasir Mohammad Khalid, Tianhao Xie, Eugene Belilovsky, and Popa Tiberiu. Clip-mesh: Generating textured meshes from text using pretrained image-text models. *SIGGRAPH Asia 2022 Conference Papers*, 2022. 3, 6

[24] Oliver Laric. Three d scans. https://threedscans.com/, 2022. 9

[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. *arXiv preprint arXiv:2211.10440*, 2022. 1, 3

[26] Jianye Lu, Athinodoros S Georghiades, Andreas Glaser, Hongzhi Wu, Li-Yi Wei, Baining Guo, Julie Dorsey, and Holly Rushmeier. Context-aware textures. *ACM Transactions on Graphics (TOG)*, 26(1):3–es, 2007. 2

[27] Tom Mertens, Jan Kautz, Jiawen Chen, Philippe Bekaert, and Frédo Durand. Texture transfer using geometry correlation. *Rendering Techniques*, 273(10.2312):273–284, 2006. 2

[28] Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. *arXiv preprint arXiv:2211.07600*, 2022. 1, 3, 6, 7

[29] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and mathematics III*, pages 35–57. Springer, 2003. 5

[30] Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2mesh: Text-driven neural stylization for meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13492–13502, 2022. 1, 3, 6, 7, 9

[31] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021. 2

[32] paffin. Teddy bear free 3d model. https://www.cgtrader.com/free-3d-models/animals/other/teddy-bear-715514f6-a1ab-4aae-98d0-80b5f55902bd, 2019. Model ID: 2034275. 7, 9

[33] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 1, 3, 6

[34] Xuebin Qin, Zichen Zhang, Chenyang Huang, Masood Dehghan, Osmar R Zaiane, and Martin Jagersand. U2-net: Going deeper with nested u-structure for salient object detection. *Pattern recognition*, 106:107404, 2020. 5

[35] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 2

[36] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. 2

[37] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. *ICCV*, 2021. 2

[38] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022. 1, 2, 3

[39] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. *arXiv preprint arXiv:2208.12242*, 2022. 2, 5, 8

[40] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding, 2022. 2

[41] Laura Salas. Klein bottle 2. https://sketchfab.com/3d-models/klein-bottle-2-eecfee26bbe54bfc9cb8e1904f33c0b7, 2016. 9

[42] savagerus. Orangutan free 3d model. https://www.cgtrader.com/free-3d-models/animals/mammal/orangutan-fe49896d-8c60-46d8-9ecb-36afa9af49f6, 2019. Model ID: 2142893. 9

[43] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022. 2

[44] Matan Sela, Yonathan Aflalo, and Ron Kimmel. Computational caricaturization of surfaces. *Computer Vision and Image Understanding*, 141:1–17, 2015. 5

[45] Omry Sendik and Daniel Cohen-Or. Deep correlations for texture synthesis. *ACM Transactions on Graphics (TOG)*, 36:1 – 15, 2017. 2

[46] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 3

[47] Yizhi Song, Zhifei Zhang, Zhe Lin, Scott Cohen, Brian Price, Jianming Zhang, Soo Ye Kim, and Daniel Aliaga. Objectstitch: Generative object compositing. *arXiv preprint arXiv:2212.00932*, 2022. 2

[48] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. *arXiv preprint arXiv:2211.12572*, 2022. 2

[49] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 6, 7, 9

[50] Wenqi Xian, Patsorn Sangkloy, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. Texturegan: Controlling deep image synthesis with texture patches. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8456–8465, 2017. 2

[51] Jiale Xu, Xintao Wang, Weihao Cheng, Yan-Pei Cao, Ying Shan, Xiaohu Qie, and Shenghua Gao. Dream3d: Zero-shot

text-to-3d synthesis using 3d shape prior and text-to-image diffusion models. *arXiv preprint arXiv:2212.14704*, 2022. 1

[52] Jonathan Young. Xatlas: Mesh parameterization / uv unwrapping library. https://github.com/jpcy/xatlas, 2022. 3

[53] Yang Zhou, Zhen Zhu, Xiang Bai, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Non-stationary texture synthesis by adversarial expansion. *arXiv preprint arXiv:1805.04487*, 2018. 2

[54] Song Chun Zhu, Yingnian Wu, and David Mumford. Filters, random fields and maximum entropy (frame): Towards a unified theory for texture modeling. *International Journal of Computer Vision*, 27(2):107–126, 1998. 2

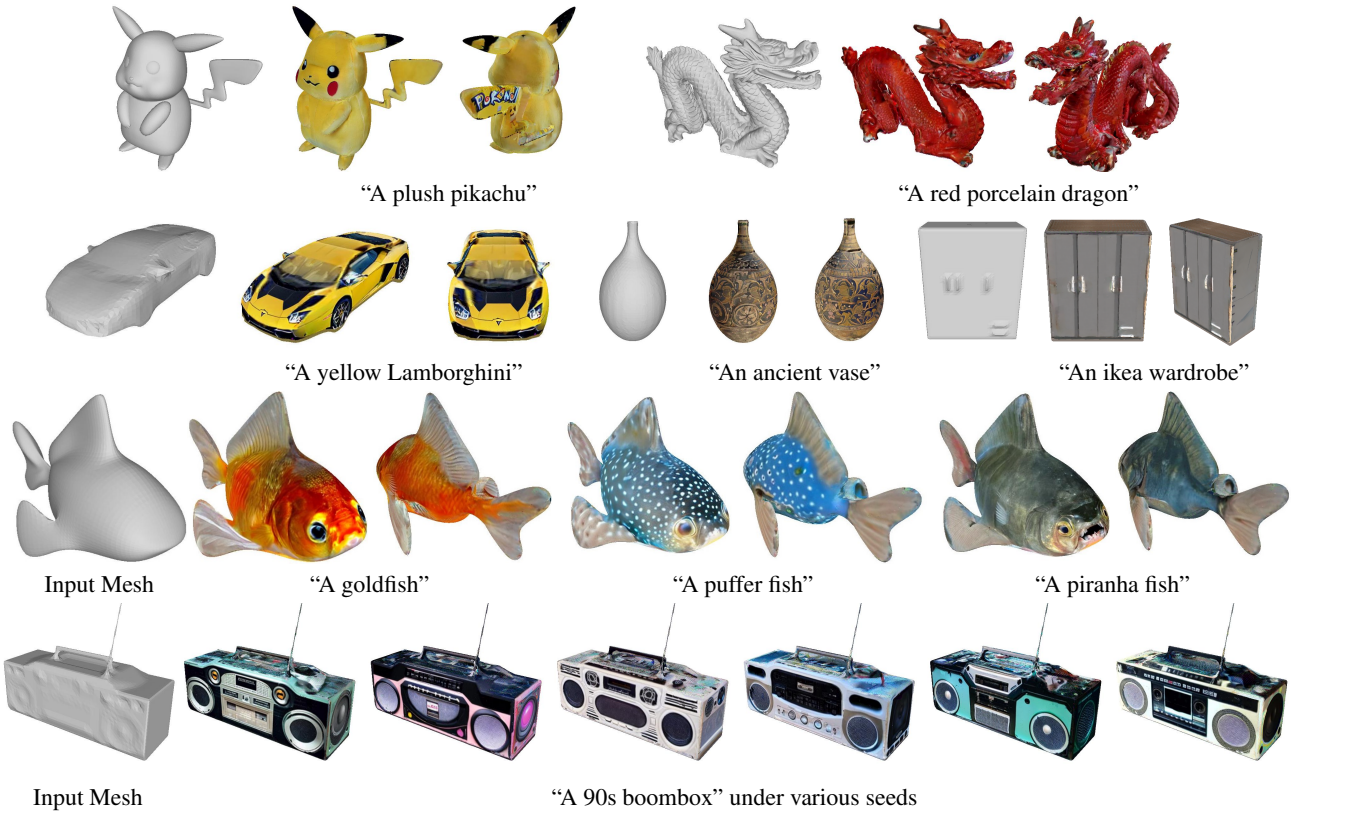

Figure 11. Additional texturing results achieved with TEXTure.

Figure 12. Additional qualitative comparison for text-guided texture generation. Best viewed zoomed in.

Figure 13. Additional texturing results achieved with TEXTure. Our method generates a high-quality texture for a collection of prompts and geometries.

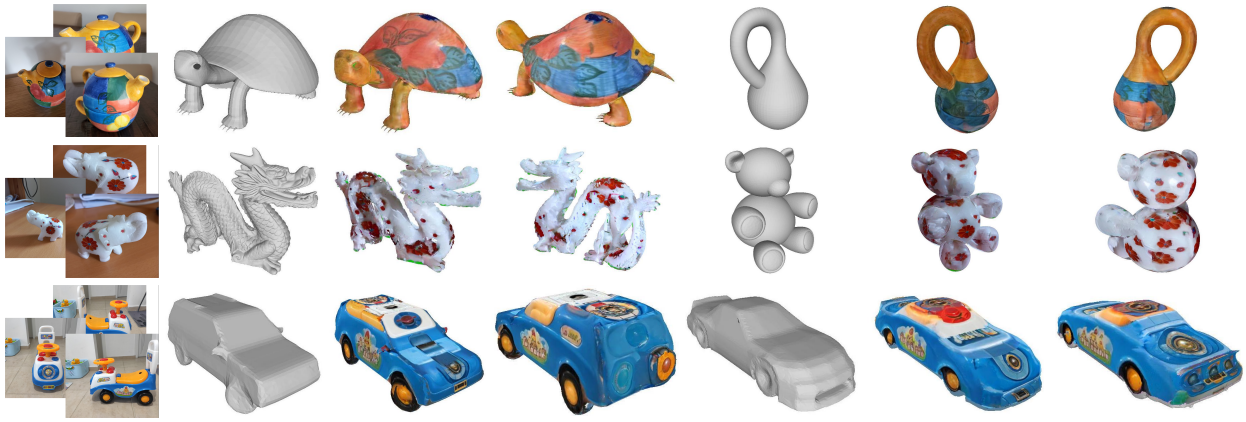

Figure 14. Token-based Texture Transfer from images. All meshes are textured with the **exact** prompt “A photo of a $\langle S_* \rangle$ using the fine-tuned diffusion model.

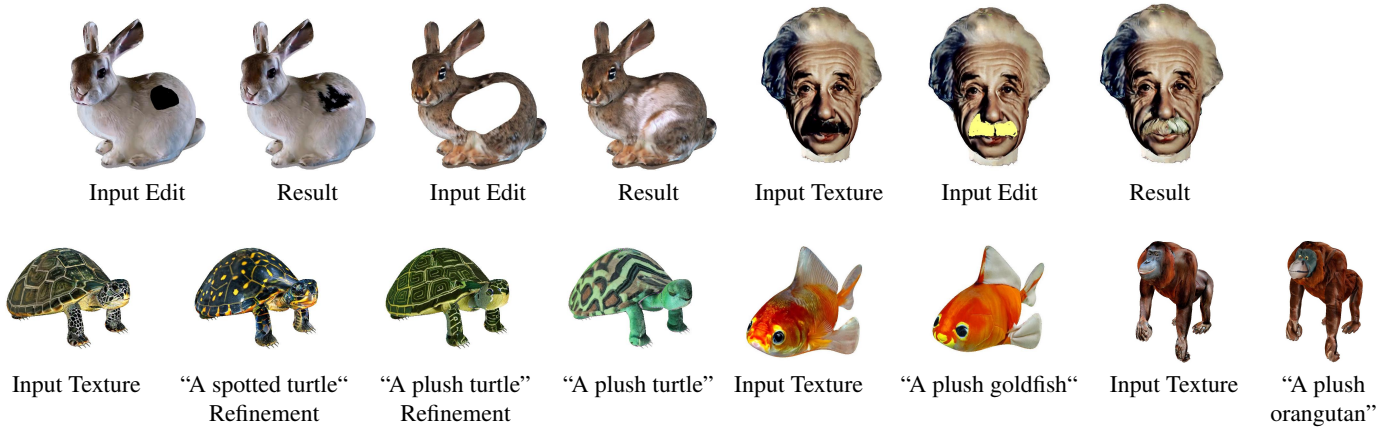

Figure 15. Texture Editing. The first row presents results for localized scribble-based editing, the original prompt was also used for the refinement step. The second row shows global text-based edits, with the last result showing a texture generated using the same prompt without conditioning on the input texture which clearly results in a completely new texture.