Text-guided Controllable Mesh Refinement for Interactive 3D Modeling
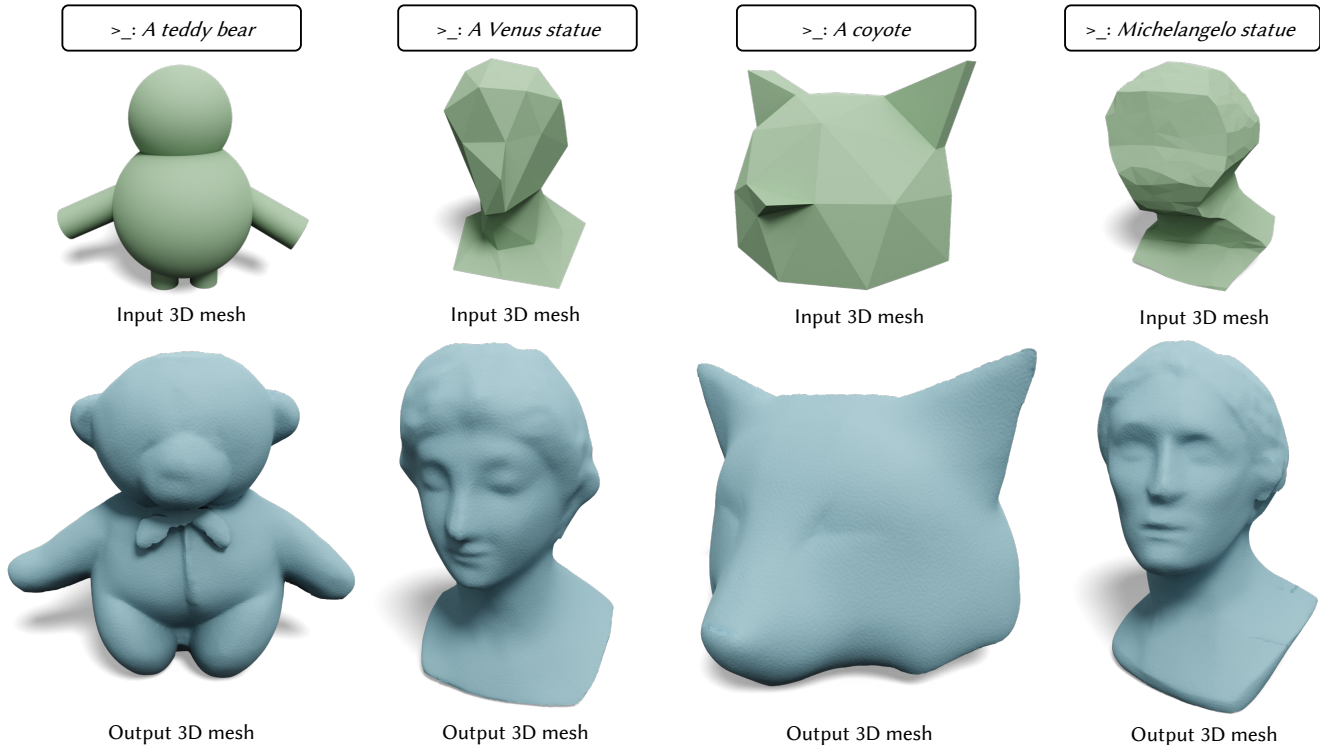
ANONYMOUS AUTHOR(S)

Fig. 1. We present a method that adds geometric details to an input (coarse) 3D mesh through text guidance. Our method can be applied to different types of input conditions. From left to right, the input mesh is an assembly of six primitive shapes, a low-poly mesh, another low-poly mesh, and a mesh initialized by silhouette carving [Kutulakos and Seitz 2000].

We propose a novel technique for adding geometric details to an input coarse 3D mesh guided by a text prompt. Our method is composed of three stages. First, we generate a single-view RGB image conditioned on the input coarse geometry and the input text prompt. This single-view image generation step allows the user to pre-visualize the result and offers stronger conditioning for subsequent multi-view generation. Second, we use our novel multi-view normal generation architecture to jointly generate six different views of the normal images. The joint view generation reduces inconsistencies and leads to sharper details. Third, we optimize our mesh with respect to all views and generate a fine, detailed geometry as output. The resulting method produces an output within seconds and offers explicit user control over the coarse structure, pose, and desired details of the resulting 3D mesh. Project page: https://text-mesh-refinement.github.io

1 INTRODUCTION

We consider the problem of adding geometric details to 3D object meshes through text guidance. Text-to-3D generative modeling has been successfully applied to computer vision [Poole et al. 2023], computer graphics [Khalid et al. 2022] and geometry processing [Gao et al. 2023] applications. These methods focus on generating a 3D mesh directly from text [Poole et al. 2023; Wang et al. 2023] without providing users with the ability to control the coarse structure of the

output shape, limiting their practical usefulness for artists who need careful control over the generation process. Other methods focus on generating textures for 3D meshes [Richardson et al. 2023], but they do not perform any changes in the geometry. While there are some approaches [Gao et al. 2023; Metzer et al. 2023] that provide control to users and are able to modify the geometry of a given shape, these methods are generally slow since they rely on costly score distillation sampling [Poole et al. 2023].

In this paper, we present a method that creates 3D object meshes with rich geometric details while allowing users to retain control over both the global shape structure (via an input coarse mesh) and the local geometric details (via an input text prompt). Following the recent success of text-guided generative methods [Metzer et al. 2023; Poole et al. 2023; Richardson et al. 2023; Wang et al. 2023], we build our method on large pre-trained text-to-image models [Rombach et al. 2022] and use language to guide the generation of geometric details. Our formulation does not require paired coarse-geometry and fine-geometry training data but instead uses large pre-trained text-to-image models as supervision to guide the process of adding geometric details through a differentiable renderer. Our key insight stems from the fact that models trained to perform text-to-image

generation guided by depth information [Mou et al. 2024; Zhang et al. 2023] end up creating images that contain additional geometric cues. As we can see in Figure 2, those cues are so prominent that even off-the-shelf normal estimation models can extract them; i.e., even though the mouse image was generated from just three spheres, its normal estimation (rightmost image) shows normals corresponding to a surface that depicts eyes, a nose and ears. However, this process is only capable of creating details visible from a single viewpoint whereas we want to add details to the whole visible surface of a given shape.

Our method adds geometric details to an input mesh in three stages. The first stage generates a single-view RGB image based on the input text prompt and the input coarse mesh. This RGB image can be seen as a preview of how the geometric details that will be added to the input mesh look like. The second stage performs multi-view generation based on the single-view output from the first stage and the input coarse mesh. The third stage refines the geometric details of the input mesh based on the multi-view generation in the second stage. Since the output of each stage is non-abstract, human-understandable images or renderings, this property allows early moments before finish for users to decide to change parameters or go back and change/modify inputs. In addition, the first two stages only involve running inferences of pre-trained networks and the final stage operates directly on meshes. Each stage can be done within seconds, therefore allowing our method to be used to support interactive 3D modeling applications. As shown in Figure 1, our method can be used to add geometric details to objects assembled by primitive shapes, to low-poly meshes, and to meshes created from silhouette carving [Kutulakos and Seitz 2000].

We compare our method with text-to-3D methods that allow for utilizing an input shape as guidance, i.e., Latent-NeRF [Metzer et al. 2023], Fantasia3D [Chen et al. 2023a] and Magic3D [Lin et al. 2023]. Extensive experimental results show that our method produces shapes that have better geometric details and visual quality based on subjective human evaluations. Our results are also more consistent with input text conditions based on both subjective human evaluations and the objective CLIP [Radford et al. 2021] similarity metric. At the same time, our method runs at least 90× faster than the competing methods.

2 RELATED WORK

Our work aims to generate shapes with rich geometric details from given coarse shapes while utilizing text prompts to guide the appearance of the generated details. Therefore, our work is closely related to methods that generate geometric details and text-to-3D generative methods. For text-to-3D methods, we focus on methods that utilize large pre-trained text-to-image diffusion models, which mainly include methods that adopt score distillation sampling (SDS) and methods based on multi-view image generation.

Geometric detail generation. With the polygon mesh representation, geometric details are typically generated on a coarse shape via transferring displacement maps or geometric texture patches from detailed shapes [Berkiten et al. 2017; Hertz et al. 2020; Zhou et al. 2006]. In addition, mesh subdivision powered by neural networks can transfer geometric details by learning to reconstruct local
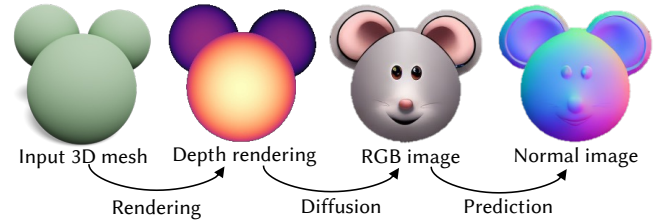

Fig. 2. **Insight.** Using a depth/normal image as a control mechanism in large text-to-image models usually leads to images that contain more details than the ones present in the original shape. Those details are so prominent that can even be captured by off-the-shelf shape estimation models.

details from training examples [Chen et al. 2023c; Liu et al. 2020]. On the other hand, methods with voxel or other grid-structured representations are able to generate geometric details by replicating local patches from reference shapes [Chen et al. 2023b, 2021; Shen et al. 2021] via patch-wise generative adversarial networks [Isola et al. 2017]. These methods typically require detailed 3D shapes as geometric style references, which limits their capability when 3D shapes are scarce. The progression on mesh-based differentiable rendering also enabled surface detail synthesis conditioned on the style of a reference image [Liu et al. 2018] or input text [Gao et al. 2023; Michel et al. 2022]. However, these methods are slow due to per-shape optimization supervised by image diffusion models [Gao et al. 2023; Michel et al. 2022].

Text-to-3D with SDS. The recent progress in text-to-image generative modeling has enabled a line of research works making use of large pre-trained text-to-image diffusion models to guide 3D generation through SDS, pioneered by DreamFusion [Poole et al. 2023] that optimizes a neural radiance field (NeRF) [Mildenhall et al. 2021] for text-to-3D generation. Several follow-up works explored optimizing NeRFs [Metzer et al. 2023; Tang et al. 2023b] and other representations with SDS, such as neural SDF [Tsalicoglou et al. 2024], meshes [Chen et al. 2023a], 3D Gaussian Splatting [Tang et al. 2023a], and hybrid representations [Lin et al. 2023; Qian et al. 2024]. Optimization-based methods re-train a model for each new text prompt, therefore suffering from a long inference time. Recent work addressed this issue by training a feed-forward model while using SDS to supervise the training [Lorraine et al. 2023].

Text-to-3D with multi-view images. Another line of works fine-tuned image diffusion models on large 3D object datasets, such as Objaverse [Deitke et al. 2023], to arrive at diffusion models capable of generating multiple images of the same 3D object viewed from different viewpoints. Such methods can produce multi-view images from a text prompt or a single-view image that could be generated by a text-to-image model. A 3D shape can then be reconstructed from the multi-view images, either via optimizing a 3D shape with differentiable rendering [Liu et al. 2023a, 2024a; Long et al. 2024; Shi et al. 2023, 2024], or via a feed-forward 3D shape reconstruction network [Hong et al. 2023; Li et al. 2023; Liu et al. 2024b, 2023b].

Although prior text-to-3D methods are expressive in the final 3D generations, most of these methods lack any control mechanism besides the input text. The unconstrained nature of these methods makes it difficult for 3D artists to incorporate them into their
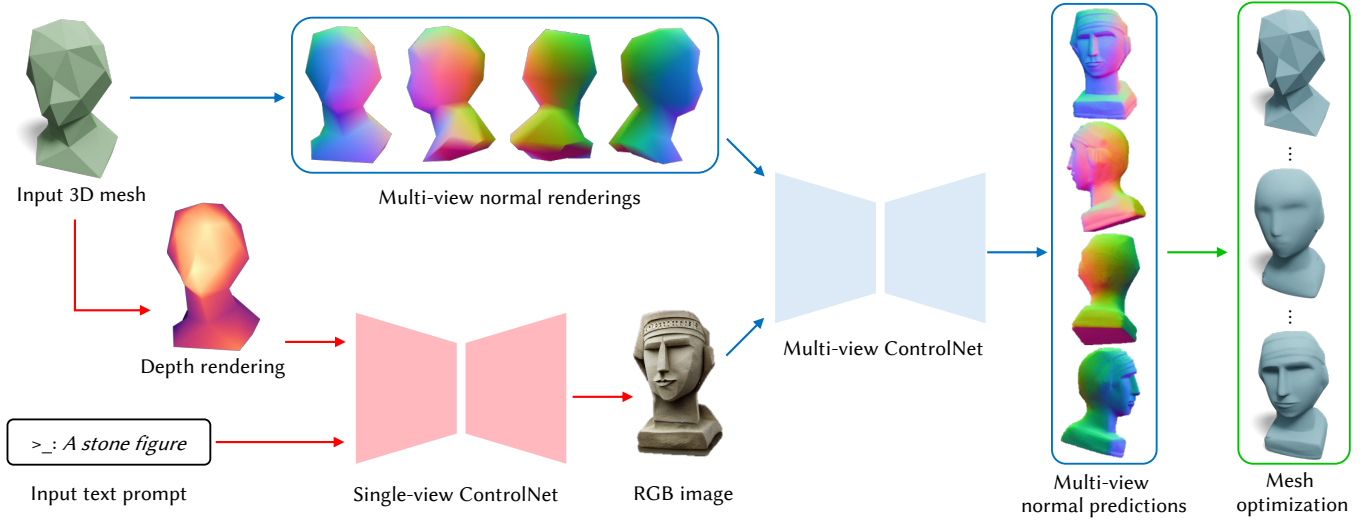
Fig. 3. **Method overview.** Our method consists of three stages: single-view generation, multi-view generation and mesh refinement/optimization. Given an input mesh and an input text prompt, we first use a large-scale pre-trained diffusion model (highlighted in red) to generate an RGB image that respects the input conditions. Next, we use a multi-view diffusion model (highlighted in blue) that takes as input the generated RGB image and the normal renderings of the input mesh and generates multi-view normals. Finally, we use the generated multi-view normals to supervise the refinement of the input mesh.

modeling process, which usually relies on fine-grained modifications. Methods that allow a single-view image as an input could provide controllability to the 3D generative process to some extent. However, a single image is often ambiguous and lacks the ability to provide sufficient information to guide the generation of the desired structure. Some optimization-based text-to-3D methods [Chen et al. 2023a; Metzer et al. 2023] allow having a coarse 3D shape to be the starting point of their generative procedure, therefore achieving generation with shape guidance. Yet they either produce poor geometry, as shown in Figure 5 (b) and (d), or deviate from the structure of the input coarse shape, as in Figure 5 (c). In addition, all methods take at least half an hour and up to several hours to generate an output due to their optimization-based nature. In contrast, we devise a multi-view ControlNet [Zhang et al. 2023] conditioned on multi-view normal renderings of the coarse input shape, to provide global structure guidance. Since our work does not rely on iterative optimization with image diffusion models, but instead relies on fast multi-view 3D reconstruction techniques [Palfinger 2022], we are able to achieve significantly faster inference speed.

3 METHOD

Our method consists of three main stages: single-view RGB generation (Section 3.1), multi-view normal generation (Section 3.2), and mesh refinement and optimization (Section 3.3). Figure 3 shows the overview of our method.

Given a coarse triangle mesh $M_{\text{coarse}} = (V_{\text{coarse}}, F_{\text{coarse}})$ with vertex positions V_{coarse} and faces F_{coarse} and a text prompt P as input, our goal is to use large pre-trained text-to-image models to generate images that have rich details and use them to refine the coarse mesh to create a triangle mesh $M_{\text{fine}} = (V_{\text{fine}}, F_{\text{fine}})$ that has additional geometric details as output. Throughout this procedure, our method will rely on a rendering operator $\mathcal{R}$ that, given a mesh

M and a viewpoint θ, yields a depth image $\mathcal{R}_d[M, \theta]$ and a normal image $\mathcal{R}_n[M, \theta]$. We will also use the generative operators $\mathcal{G}$ that, given a control image I and a text prompt P, output a new image $\mathcal{G}[I, P]$ that follows both I's structure and P's description. In practice, those operators correspond to deep generative models like ControlNet [Zhang et al. 2023] and T2I-Adapters [Mou et al. 2024].

3.1 Single-View RGB Generation

The goal of the first stage of our approach is to generate an RGB image that will act as a guide for the rest of the mesh refinement process. This is necessary for two reasons. First, this image can be generated using large-scale text-to-image models that are general enough to work with various prompts and types of objects while still being capable of creating detailed imagery. Second, this initial procedure can act as a fast preview of the whole process – the user can quickly verify (in ~3 seconds) if the image corresponds to the appearance they intend before continuing with the rest of the pipeline. The process starts by the user providing a starting viewpoint θ_s, an initial mesh M_{coarse} and the text prompt P. We use a standard depth-to-RGB ControlNet model $\mathcal{G}_d$ to create a color image I_s that will be used in the rest of the process:

$$I_s = \mathcal{G}_d\big[\mathcal{R}_d[M_{\text{coarse}}, \theta_s], P\big]. \tag{1}$$

An important feature of using $\mathcal{G}_d$ to create I_s is that its generative process happens through a number of backward diffusion steps, i.e., iteratively denoising an image whose pixels are sampled from $\mathcal{N}(0, 1)$. Thus, we can allow the user to control how much I_s deviates from $\mathcal{R}_d[M_{\text{coarse}}, \theta_s]$ by disabling the depth guidance in the later iterations of the denoising procedure. We refer readers to Zhang et al. [2023] for further details about the model architecture and how the amount of guidance can be set by the user. In Section 4.2 we discuss

the effects of applying different levels of guidance throughout the single-view and multi-view generation stages.

After this process is complete, we have an RGB image I_s that roughly follows the structure given by M_{coarse} when seen from θ_s while depicting the description P. Unfortunately, I_s is generated based on a single-view depth rendering of the input mesh and has no information about what the mesh we are trying to refine looks like from viewpoints other than θ_s. One could use other methods that generate multi-view consistent images from a single input image (e.g., MVDream [Shi et al. 2024] and Wonder3D [Long et al. 2024]), but they would not follow M_{coarse} – only what it looks like when seen from θ_s. Examples of this behavior are depicted in Figure 6 and Figure 11. While Wonder3D and other image-to-3D methods are capable of generating reasonable shapes, they fail to adhere to any shape guidance not depicted in the input image. We address this issue by designing a multi-view ControlNet architecture detailed in the following subsection.

3.2 Multi-view Normal Generation

Until this point, we only used guidance from the initial viewpoint θ_s. Now, we define a set of viewpoints $\Theta = \{R_y(\alpha)|\alpha \in \{0, \frac{\pi}{4}, \frac{\pi}{2}, \pi, \frac{3\pi}{2}, \frac{7\pi}{4}\}\}$ where $R_y(\alpha)$ corresponds to the viewpoint obtained by rotating the object α degrees along the upward axis from the starting viewpoint θ_s. We propose a multi-view ControlNet $\mathcal{G}^{\text{mv}}$ that learns to generate multi-view consistent normal images $\{I_{n,\theta}\}_{\theta \in \Theta}$ conditioned on the single-view RGB image I_s and on $|\Theta|$ multi-view normal renderings $\{\mathcal{R}_n[M_{\text{coarse}}, \theta]\}_{\theta \in \Theta}$ of the input mesh M_{coarse}. More precisely, we have

$$I_{n,\theta} = \mathcal{G}_{n,\theta}^{\text{mv}}[\{\mathcal{R}_n[M_{\text{coarse}}, \theta]\}_{\theta \in \Theta}, I_s], \quad (2)$$

where $\mathcal{G}_{n,\theta}^{\text{mv}}$ yields normal images of the object seen from θ. Thanks to the additional information present in I_s and the way $\mathcal{G}^{\text{mv}}$ is trained, the normal images $I_{n,\theta}$ generated by our multi-view ControlNet preserve the coarse structure of M_{coarse} while containing details following the description P. We build our $\mathcal{G}^{\text{mv}}$ on top of a Wonder3D [Long et al. 2024] network $\mathcal{W}^{\text{mv}}$ that is trained to generate multi-view consistent RGB and normal images given a single-view RGB image. We take inspiration from ControlNet [Zhang et al. 2023] and design a control model C^{mv} that can be plugged into $\mathcal{W}^{\text{mv}}$ to allow multi-view control in the existent multi-view generation network.

Network details. $\mathcal{W}^{\text{mv}}$ builds on pre-trained 2D Stable Diffusion [Rombach et al. 2022] and enables multi-view RGB and normal generation from a single-view RGB image. The network takes as input a single-view RGB image and a camera pose, and outputs an RGB image and a normal image of the object depicted in the input image from the given camera pose. We refer readers to Long et al. [2024] for details about Wonder3D. Our model $\mathcal{G}^{\text{mv}}$ consists of three components: the original (frozen) $\mathcal{W}^{\text{mv}}$, a trainable C^{mv} copy of $\mathcal{W}^{\text{mv}}$, and a normal encoder, which takes as input the set $\{\mathcal{R}_n[M_{\text{coarse}}, \theta]\}_{\theta \in \Theta}$. Specifically, C^{mv}'s encoder takes as input the output of the normal encoder and outputs features that are then integrated into $\mathcal{W}^{\text{mv}}$'s decoder's skip connections using zero-initialized convolutions. We follow the same training procedure as Wonder3D to train the normal encoder and C^{mv}.
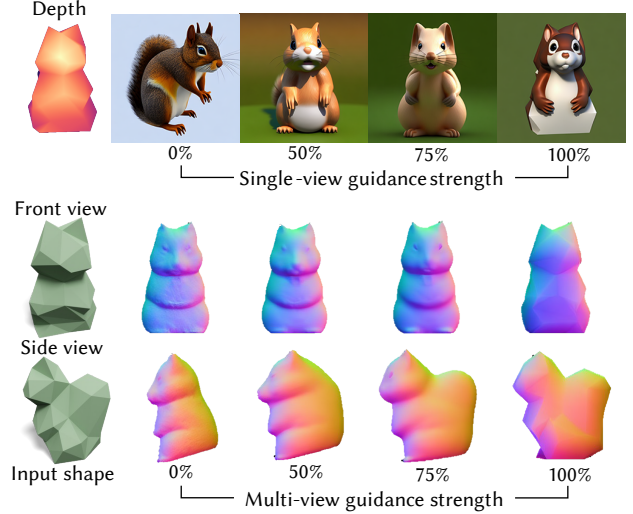

Fig. 4. **Influence of guidance strength.** The user can control how much detail can be generated by the system when setting the number of backward diffusion steps where the guidance will be used. This control is available during the single-view (top) and multi-view (bottom) stages.

Training data. To train our multi-view ControlNet, we generate training data using the LVIS subset of the Objaverse [Deitke et al. 2023] dataset, which consists of around 32,000 objects. For each object, we use Blenderproc [Denninger et al. 2023] to render RGB images and normal images from 6 views, i.e., the front, left, right, back, front-left and front-right views. To encourage our multi-view ControlNet to learn to generate more details in the multi-view normal predictions, for every input normal image we apply a Gaussian filter with a kernel size of 7. The goal of this blurring step is to ensure that the model is only aware of the coarse structure of the shape and not its specific details. To train our multi-view ControlNet, we use the blurred normal image as input to the normal encoder and train the network to predict the unblurred (clear) normal image. This mechanism allows the model to learn to generate additional geometric details while following the coarse structure of the blurred normals. This blurring operation is also applied at inference time.

Implementation details. We implement our network in PyTorch. The normal encoder consists of 5 convolutional layers. The first four convolution layers are randomly initialized, each of which is followed by a SiLU [Elfwing et al. 2018] activation. The last convolution layer is zero-initialized. The image resolution used to train the network is 256×256. We use a batch size of 192 and train the network for 20,000 steps. We use 4 NVIDIA A100 GPUs to train the network. The network training takes around 4 days.

3.3 Mesh Refinement and Optimization

Given the multi-view images generated by our multi-view ControlNet, there are multiple ways to reconstruct a 3D mesh from them. For instance, one can use multi-view images to fit a NeRF [Mildenhall et al. 2021; Müller et al. 2022] or a neural SDF [Wang et al. 2021]. However, computing these representations through gradient-based optimization takes a lot of time and, after the procedure is

>_: Santa Claus

>_: A dolphin

>_: A teddy bear

(a) Input 3D mesh   (b) Latent-NeRF   (c) Fantasia3D   (d) Magic3D   (e) Ours
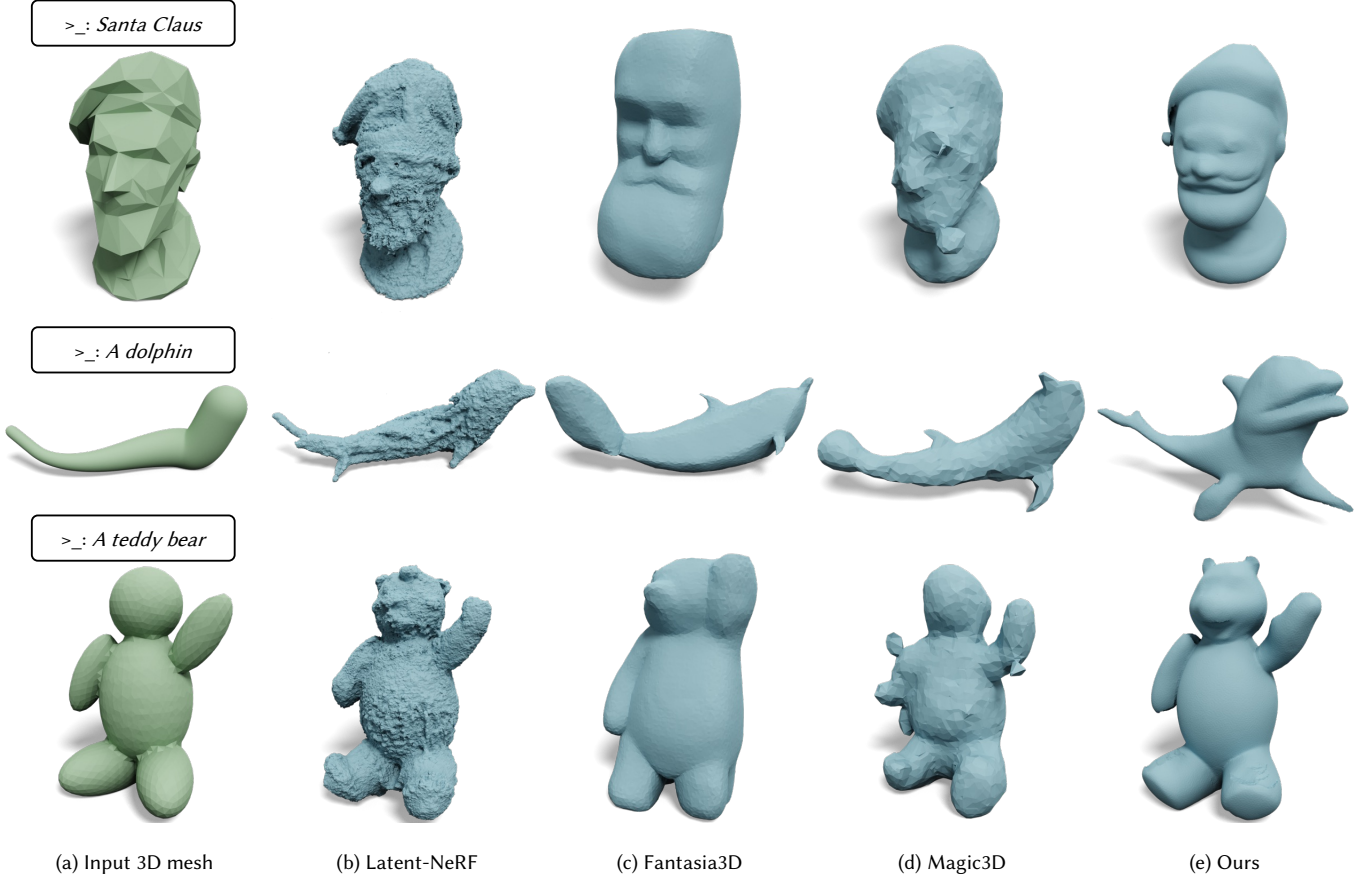
Fig. 5. **Qualitative results.** Our method generates 3D meshes that have better geometric details and visual quality compared to state-of-the-art methods.

complete, one would still need to convert them into meshes. For our application, we cast mesh refinement directly as a mesh optimization problem. To make the mesh optimization pipeline fully differentiable, we use a differentiable rasterizer, nvdiffrast [Laine et al. 2020], that allows us to quickly optimize the coarse mesh M_{coarse} to follow the set of normal images $\{I_{n,\theta}\}_{\theta \in \Theta}$.

We will optimize the mesh M initialized as $M = M_{\text{coarse}}$ and refine its geometry based on $\{I_{n,\theta}\}$ through gradient descent. For each iteration, we optimize the mesh vertices to minimize an objective function $\mathcal{L}$ with three components:

$$\mathcal{L} = \mathcal{L}_{\text{n}} + \mathcal{L}_{\nabla n} + \mathcal{L}_{\text{sil}}. \tag{3}$$

For simplicity, we will slightly abuse the notation and define $\|a - b\|_1$ as the sum of the absolute differences between all the pixels in image a and image b. Thus, $\mathcal{L}_{\text{n}}$ is the ℓ_1 loss between the normal rendering of the mesh and the normal generated by $\mathcal{G}^{\text{mv}}$:

$$\mathcal{L}_{\text{n}} = \frac{1}{|\Theta|} \sum_{\theta \in \Theta} \left\| \mathcal{R}_n[M, \theta] - I_{n,\theta} \right\|_1. \tag{4}$$

$\mathcal{L}_{\nabla n}$ is the ℓ_1 loss between the image gradient of the normal rendering of the mesh and the image gradient of the target normal:

$$\mathcal{L}_{\nabla n} = \frac{1}{|\Theta|} \sum_{\theta \in \Theta} \left\| \frac{\partial \mathcal{R}_n[M, \theta]}{\partial x} - \frac{\partial I_{n,\theta}}{\partial x} \right\|_1 + \left\| \frac{\partial \mathcal{R}_n[M, \theta]}{\partial y} - \frac{\partial I_{n,\theta}}{\partial y} \right\|_1. \tag{5}$$

We found experimentally that $\mathcal{L}_{\nabla n}$ is a crucial component for capturing the higher frequency details present in the generated normal images. See Figure 7 for an example comparing the mesh optimization procedure with and without this component. $\mathcal{L}_{\text{sil}}$ is the ℓ_1 loss between the silhouette rendering $\mathcal{R}_s$ of the mesh and the foreground segmentation mask $\mathcal{S}[I_{\text{rgb},\theta}]$ predicted from the RGB image $I_{\text{rgb},\theta}$ (generated by $\mathcal{G}^{\text{mv}}$) using Segment Anything [Kirillov et al. 2023]:

$$\mathcal{L}_{\text{sil}} = \frac{1}{|\Theta|} \sum_{\theta \in \Theta} \left\| \mathcal{R}_s[M, \theta] - \mathcal{S}[I_{\text{rgb},\theta}] \right\|_1. \tag{6}$$

After each optimization iteration, we follow continuous remeshing [Palfinger 2022] to adaptively remesh the mesh M.

Implementation details. We implement our mesh optimization algorithm in PyTorch. We run mesh optimization for 2,000 iterations. The mesh optimization process takes around 20 seconds on one single NVIDIA 3080Ti GPU.

Table 1. **Quantitative results.** Our method achieves the best results on CLIP similarity while being approximately two orders of magnitude faster.

Method	CLIP similarity ↑	Run time (sec) ↓
Magic3D	0.2327	5597
Latent-NeRF	0.2367	3697
Fantasia3D	0.2556	2971
Ours	**0.2666**	**32**

4 EXPERIMENTS

In this section, we analyze the capabilities of our method and investigate how modifying various components impacts its overall performance. We start by presenting a quantitative and qualitative comparison with the current state-of-the-art in the text-guided mesh refinement task. Then, we show how the user can apply different guidance levels to trade off between detail generation and original shape preservation. Finally, we present a few additional results and applications.

4.1 Text-guided Mesh Refinement

Baseline methods. We choose Latent-NeRF [Metzer et al. 2023], Magic3D [Lin et al. 2023] and Fantasia3D [Chen et al. 2023a] as the competing methods. Latent-NeRF introduces global shape guidance through a sketch shape and a loss between the underlying NeRF's occupancy value and the winding number to the sketch shape's surface. We use their official implementation and their default setting to generate results for comparison with our method. Fantasia3D has their shape-guided 3D generation procedure implemented by initializing their initial SDF representation with an input shape. However, they don't enforce any control mechanism during the generation. As a result, their generation can diverge from the input shape's outline. Magic3D introduced a two-stage method for text-to-3D generation. Their second stage refines the coarse shape generated from the first stage via direct mesh optimization with differentiable rendering and it is therefore comparable to our method. For a fair comparison, we remove the first stage and initialize the Magic3D's second stage geometry with the guidance shape in a randomly initialized color field. For both Magic3D and Fantasia3D, we use the public implementation by threestudio [Guo et al. 2023]. Notice that all these methods rely on the score distillation sampling which usually takes more than 30 minutes per generation while our method takes less than a minute. For a runtime comparison, please refer to Table 1.

Quantitative results. Table 1 reports the CLIP [Radford et al. 2021] similarity result and the run time of each method. We use the public implementation of the CLIP similarity metric from Zhengwentai [2023]. This metric computes the cosine similarity between the text prompt and images of the final meshes. Our method achieves the best CLIP similarity while running almost two orders of magnitude faster.

Perceptual study and qualitative results. We also present a casual perceptual study comparing our method with the three competing baselines. We designed 18 questions applied to 41 different participants. For each question, we present to the participant an input

Table 2. **User study results.** Our method was selected as the favorite for the majority of users in all evaluated setups.

Comparison	Preference
Ours vs. Magic3D	
• Better geometric details	96.35%
• Better visual quality	100.00%
• More consistent with input text	98.80%
Ours vs. Latent-NeRF	
• Better geometric details	89.05%
• Better visual quality	93.90%
• More consistent with input text	76.80%
Ours vs. Fantasia3D	
• Better geometric details	84.15%
• Better visual quality	65.85%
• More consistent with input text	69.50%

>_: A cat statue

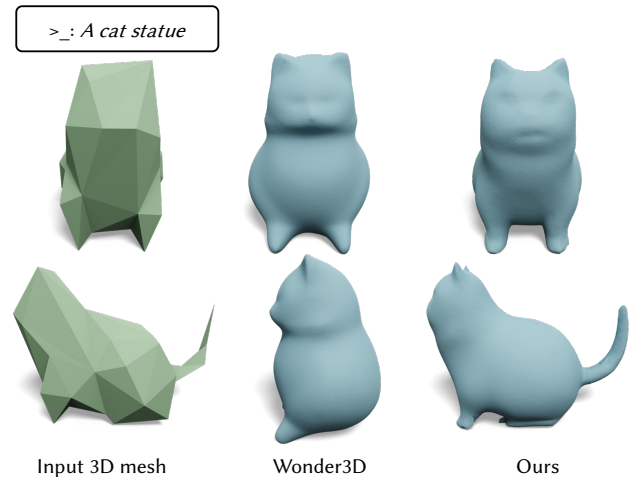Input 3D mesh   Wonder3D   Ours

Fig. 6. **Multi-view control.** Our method is capable of generating details using the full initial shape as guidance. Notice how the back legs of the cat and its tail follow the input coarse mesh (in green). On the other hand, Wonder3D results yield reasonable renditions when visualized from the initial viewpoint θ_s (top row) but clearly fail to follow the coarse geometric guidance when seen from other views (bottom row).

coarse mesh, a text prompt, a mesh refined by our method and a mesh refined by one of the three baseline methods. We ask the participant to select the mesh that has better geometric details, has better visual quality, and is more consistent with the input text prompt. Based on the results, we compute the preference, i.e., the percentage of participants who prefer the mesh refined by our method. Table 2 reports the study results. Figure 5 shows the qualitative results of each method. More visual results are provided in Figure 9 and Figure 10. The perceptual study result indicates that our method is favored across all dimensions analyzed. Qualitatively, Latent-NeRF uses a NeRF representation that notoriously has problems generating surfaces since it is fundamentally a volumetric occupancy
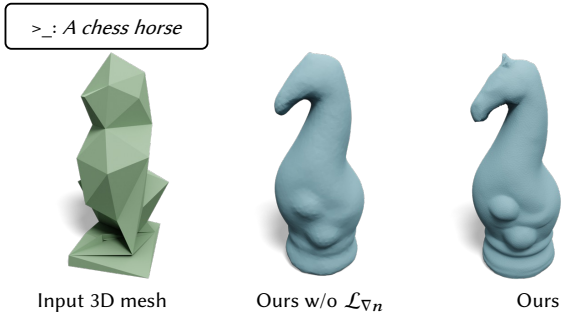
>_: A chess horse

Input 3D mesh   Ours w/o $\mathcal{L}_{\nabla n}$ Ours

Fig. 7. **Ablating** $\mathcal{L}_{\nabla n}$. Adding the $\mathcal{L}_{\nabla n}$ component during mesh optimization allows the final mesh to incorporate more details from the generated normal images. In this example, notice the ears of the horse and the crisper details on the bottom of the piece.

representation. While Magic3D also suffers from generating meaningful results. Fantasia3D is based on SDS and generates visually good results at the expense of a long run time. Our method generates results that are more consistent with the input 3D mesh structure than the other methods and is much faster since we only need to run inference on pre-trained networks. The mesh refinement step operates directly on the input mesh and converges within seconds.

4.2 Controllability

Control guidance strength. We allow the users to control how much detail is generated by allowing them to select how many percentage of the backward diffusion process will use the given control guidance. As seen in Figure 4, this can happen in both single-view and multi-view stages. During the single-view stage, not using $\mathcal{R}_d[M_{\text{coarse}}, \theta_s]$ means creating an image uncorrelated to M_{coarse}, not desirable for our application. Using the control guidance throughout the whole generative process, on the other hand, may result in an image that follows M_{coarse} too strictly which might incur in artificial shapes like the squirrel to the right in Figure 4. During the multi-view stage, the guidance will control how many percentage of the multi-view generation process uses images $\{\mathcal{R}_n[M_{\text{coarse}}, \theta]\}_{\theta \in \Theta}$ as control guidance – when not used at all (0% in the bottom row of Figure 4), our multi-view generation stage is reduced to Wonder3D. As the control guidance increases, information from other views becomes more relevant, but using control guidance 100% of the denoising steps (100% in the bottom row of Figure 4) yields a final shape that is very similar to M_{coarse} without too many added details. We empirically found that values between 30% and 75% strike a good balance between controllability and detail generation.

Pose control. Our method allows users to provide explicit control over the pose of the resulting 3D mesh. Figure 12 shows three examples, each of which has a different pose. Our method generates 3D meshes that have more geometric details and are consistent with the pose of the input 3D mesh.

4.3 Application

We develop a demo that builds on the Gradio library to provide users with a user-friendly interface for 3D modeling applications. See the demo video at https://text-mesh-refinement.github.io.
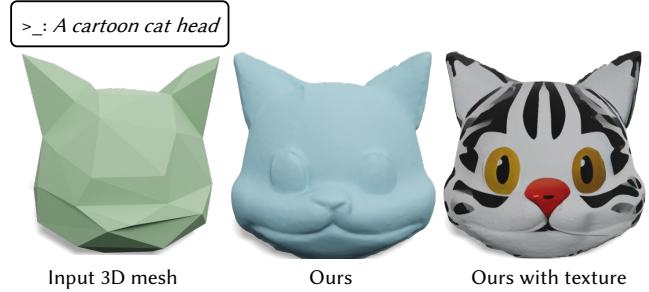

>_: A cartoon cat head

Input 3D mesh   Ours   Ours with texture

Fig. 8. **Mesh texturing.** Our final refined mesh can also be textured.

4.4 Mesh Texturing

While the goal of our method is to add geometric details to input 3D meshes, our method can also be used to texture a mesh. Since our method builds on Wonder3D which outputs both multi-view RGB images and normals, we can use the multi-view RGB predictions to texture the output 3D mesh. We use the mesh texturing code from Direct2.5 [Lu et al. 2024]. We refer readers to Lu et al. [2024] for details about mesh texturing. Figure 8 shows an example.

5 CONCLUSIONS

In this paper, we offer a novel fast method for adding high-quality geometric details to coarse 3D meshes. Our key design decision is to rely only on feed-forward networks, generating a single-view RGB image conditioned on the coarse input mesh, to further use as a condition for multi-view ControlNet that we train to generate multi-view consistent normal images. Finally, we can use the resulting normal images to refine the input coarse geometry in a matter of seconds, since we only need differentiable rasterization, and do not rely on propagating gradients via any diffusion networks. We demonstrate that our method is useful in several scenarios, including refining low-poly shapes and shapes produced from an arrangement of simple primitives, and also supports texturing the refined mesh. In addition to speed and quality, our method offers explicit control over the pose and the coarse structure of the output, which we believe is essential for real creative applications.

Limitations and future work. Our method was trained to generate 6 images of 256×256 resolution for the subsequent mesh refinement. The number of views and the image resolution limit the level of geometric details our method can generate. In addition, our mesh refinement algorithm relies on the segmentation quality of an off-the-shelf image segmentation model. The refinement produces artifacts when the segmentation is faulty or inconsistent across different views. Training a model to generate more view-consistent images with higher resolution, as well as employing more robust image segmentation models, can boost the performance of our method. There are other interesting topics raised during the development of our method, such as image-guided mesh refinement, localized detail generation in selected regions or parts, and identity preservation when changing the pose of the coarse mesh (see Figure 12), which we aim to explore in the future.

REFERENCES

Sema Berkiten, Maciej Halber, Justin Solomon, Chongyang Ma, Hao Li, and Szymon Rusinkiewicz. 2017. Learning detail transfer based on geometric features. *Computer Graphics Forum* (2017).

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 *ACM SIGGRAPH Asia*.

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

Yun-Chun Chen, Vladimir Kim, Noam Aigerman, and Alec Jacobson. 2023c. Neural Progressive Meshes. In *ACM SIGGRAPH*.

Zhiqin Chen, Vladimir G Kim, Matthew Fisher, Noam Aigerman, Hao Zhang, and Siddhartha Chaudhuri. 2021. Decor-gan: 3d shape detailization by conditional refinement. In *CVPR*.

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli Vander-Bilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. 2023. Objaverse: A universe of annotated 3d objects. In *CVPR*.

Maximilian Denninger, Dominik Winkelbauer, Martin Sundermeyer, Wout Boerdijk, Markus Wendelin Knauer, Klaus H Strobl, Matthias Humt, and Rudolph Triebel. 2023. Blenderproc2: A procedural pipeline for photorealistic rendering. *Journal of Open Source Software* (2023).

Stefan Elfwing, Eiji Uchibe, and Kenji Doya. 2018. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural networks* (2018).

William Gao, Noam Aigerman, Thibault Groueix, Vova Kim, and Rana Hanocka. 2023. Textdeformer: Geometry manipulation using text guidance. In *ACM SIGGRAPH*.

Yuan-Chen Guo, Ying-Tian Liu, Ruizhi Shao, Christian Laforte, Vikram Voleti, Guan Luo, Chia-Hao Chen, Zi-Xin Zou, Chen Wang, Yan-Pei Cao, and Song-Hai Zhang. 2023. threestudio: A unified framework for 3D content generation. https://github.com/threestudio-project/threestudio.

Amir Hertz, Rana Hanocka, Raja Giryes, and Daniel Cohen-Or. 2020. Deep geometric texture synthesis. *ACM Transactions on Graphics* (2020).

Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. 2023. LRM: Large Reconstruction Model for Single Image to 3D. In *ICLR*.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *CVPR*.

Nasir Mohammad Khalid, Tianhao Xie, Eugene Belilovsky, and Tiberiu Popa. 2022. Clip-mesh: Generating textured meshes from text using pretrained image-text models. In *ACM SIGGRAPH Asia*.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. In *ICCV*.

Kiriakos N Kutulakos and Steven M Seitz. 2000. A theory of shape by space carving. *IJCV* (2000).

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* (2020).

Jiahao Li, Hao Tan, Kai Zhang, Zexiang Xu, Fujun Luan, Yinghao Xu, Yicong Hong, Kalyan Sunkavalli, Greg Shakhnarovich, and Sai Bi. 2023. Instant3D: Fast Text-to-3D with Sparse-view Generation and Large Reconstruction Model. In *ICLR*.

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 *CVPR*.

Hsueh-Ti Derek Liu, Vladimir G Kim, Siddhartha Chaudhuri, Noam Aigerman, and Alec Jacobson. 2020. Neural subdivision. *ACM Transactions on Graphics* (2020).

Hsueh-Ti Derek Liu, Michael Tao, and Alec Jacobson. 2018. Paparazzi: surface editing by way of multi-view image processing. *ACM Transactions on Graphics* (2018).

Minghua Liu, Ruoxi Shi, Linghao Chen, Zhuoyang Zhang, Chao Xu, Xinyue Wei, Hansheng Chen, Chong Zeng, Jiayuan Gu, and Hao Su. 2024b. One-2-3-45++: Fast single image to 3d objects with consistent multi-view generation and 3d diffusion. In *CVPR*.

Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Mukund Varma T, Zexiang Xu, and Hao Su. 2023b. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. In *NeurIPS*.

Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. 2023a. Zero-1-to-3: Zero-shot one image to 3d object. In *ICCV*.

Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. 2024a. Syncdreamer: Generating multiview-consistent images from a single-view image. In *ICLR*.

Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, et al. 2024. Wonder3d: Single image to 3d using cross-domain diffusion. In *CVPR*.

Jonathan Lorraine, Kevin Xie, Xiaohui Zeng, Chen-Hsuan Lin, Towaki Takikawa, Nicholas Sharp, Tsung-Yi Lin, Ming-Yu Liu, Sanja Fidler, and James Lucas. 2023. Att3d: Amortized text-to-3d object synthesis. In *ICCV*.

Yuanxun Lu, Jingyang Zhang, Shiwei Li, Tian Fang, David McKinnon, Yanghai Tsin, Long Quan, Xun Cao, and Yao Yao. 2024. Direct2. 5: Diverse text-to-3d generation via multi-view 2.5 d diffusion. In *CVPR*.

Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. 2023. Latent-nerf for shape-guided generation of 3d shapes and textures. In *CVPR*.

Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. 2022. Text2Mesh: Text-driven neural stylization for meshes. In *CVPR*.

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* (2021).

Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, and Ying Shan. 2024. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. In *AAAI*.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics* (2022).

Werner Palfinger. 2022. Continuous remeshing for inverse rendering. *Computer Animation and Virtual Worlds* (2022).

Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. 2023. Dreamfusion: Text-to-3d using 2d diffusion. In *ICLR*.

Guocheng Qian, Jinjie Mai, Abdullah Hamdi, Jian Ren, Aliaksandr Siarohin, Bing Li, Hsin-Ying Lee, Ivan Skorokhodov, Peter Wonka, Sergey Tulyakov, et al. 2024. Magic123: One image to high-quality 3d object generation using both 2d and 3d diffusion priors. In *ICLR*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *ICML*.

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

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

Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. 2021. Deep Marching Tetrahedra: a hybrid representation for high-resolution 3D shape synthesis. In *NeurIPS*.

Ruoxi Shi, Hansheng Chen, Zhuoyang Zhang, Minghua Liu, Chao Xu, Xinyue Wei, Linghao Chen, Chong Zeng, and Hao Su. 2023. Zero123++: a single image to consistent multi-view diffusion base model. *arXiv* (2023).

Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. 2024. Mvdream: Multi-view diffusion for 3d generation. In *ICLR*.

Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. 2023a. DreamGaussian: Generative Gaussian Splatting for Efficient 3D Content Creation. In *ICLR*.

Junshu Tang, Tengfei Wang, Bo Zhang, Ting Zhang, Ran Yi, Lizhuang Ma, and Dong Chen. 2023b. Make-it-3d: High-fidelity 3d creation from a single image with diffusion prior. In *ICCV*.

Christina Tsalicoglou, Fabian Manhardt, Alessio Tonioni, Michael Niemeyer, and Federico Tombari. 2024. Textmesh: Generation of realistic 3d meshes from text prompts. In *3DV*.

Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. 2021. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *NeurIPS*.

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 *NeurIPS*.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023. Adding conditional control to text-to-image diffusion models. In *ICCV*.

SUN Zhengwentai. 2023. clip-score: CLIP Score for PyTorch. https://github.com/taited/clip-score.

Kun Zhou, Xin Huang, Xi Wang, Yiying Tong, Mathieu Desbrun, Baining Guo, and Heung-Yeung Shum. 2006. Mesh quilting for geometric texture synthesis. In *ACM SIGGRAPH*.
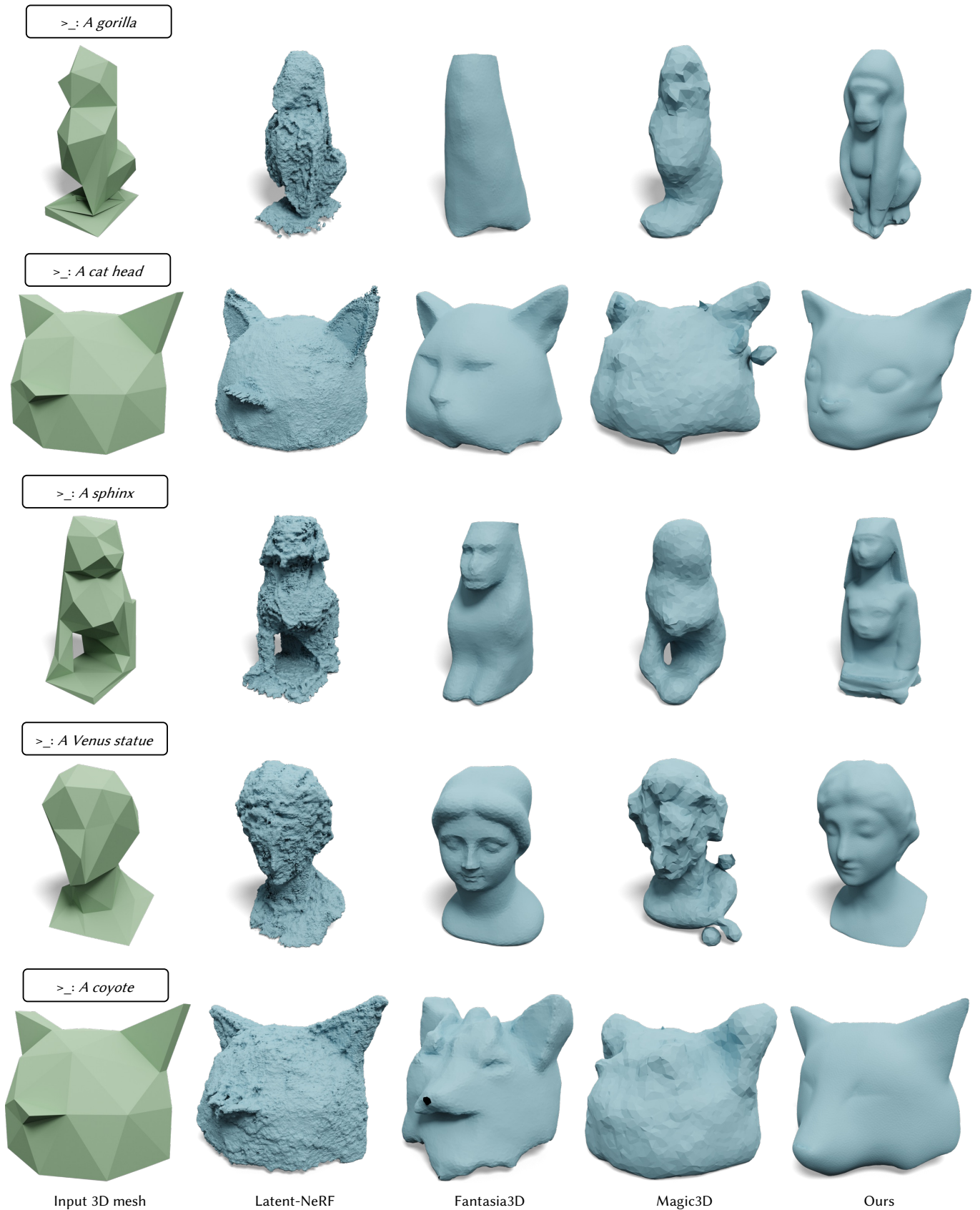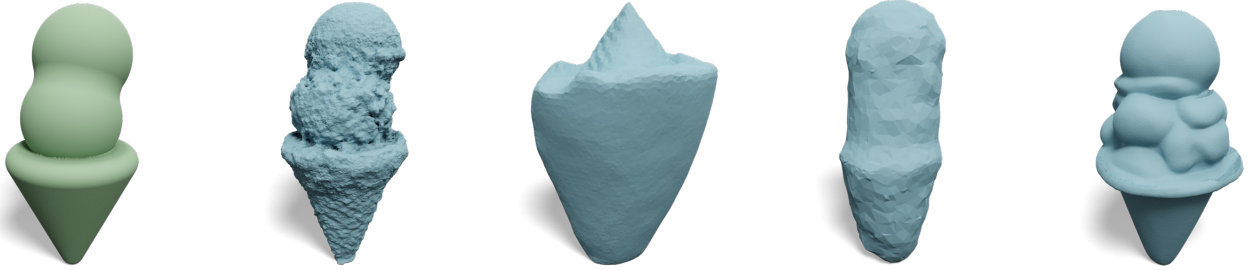
Fig. 9. **Qualitative results.** Our method generates 3D meshes that have better geometric details and visual quality compared to state-of-the-art methods.

>_: A delicious ice cream

>_: A cartoon pig head
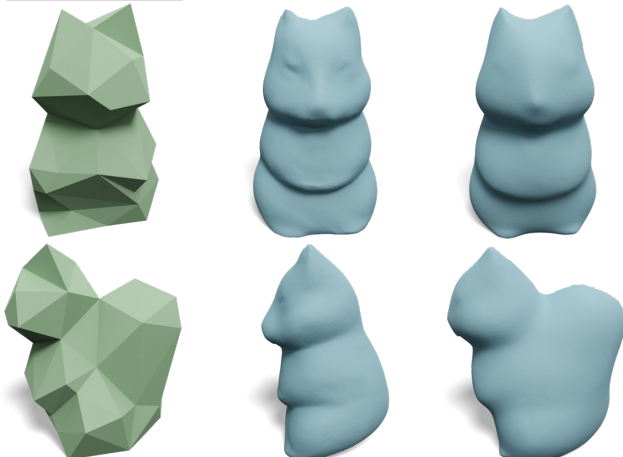
>_: Michelangelo statue

| Input 3D mesh | Latent-NeRF | Fantasia3D | Magic3D | Ours |

Fig. 10. **Qualitative results.** Our method generates 3D meshes that have better geometric details and visual quality compared to state-of-the-art methods.

>_: A cat statue

>_: A cartoon figure   >_: A teddy bear   >_: A teddy bear

| Input 3D mesh | Wonder3D | Ours |

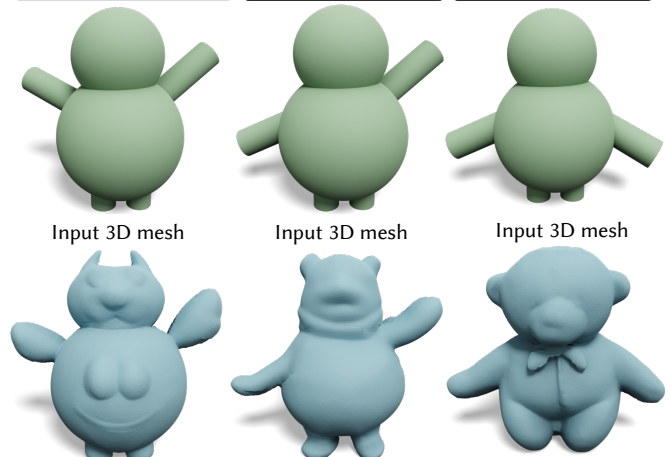| Input 3D mesh | Input 3D mesh | Input 3D mesh |
| Output 3D mesh | Output 3D mesh | Output 3D mesh |

Fig. 11. **Multi-view control.** Our method is capable of generating details using the full initial shape as guidance. On the other hand, Wonder3D results yield reasonable renditions when visualized from the initial viewpoint θ_s (top row) but clearly fail to follow the coarse geometric guidance when seen from other views (bottom row).

Fig. 12. **Pose control.** Our method is capable of generating meshes that follow the pose of the input mesh.