Magic3D: High-Resolution Text-to-3D Content Creation

Chen-Hsuan Lin* Jun Gao* Luming Tang* Towaki Takikawa* Xiaohui Zeng*
Xun Huang Karsten Kreis Sanja Fidler† Ming-Yu Liu† Tsung-Yi Lin
NVIDIA Corporation

Abstract

DreamFusion [33] has recently demonstrated the utility of a pre-trained text-to-image diffusion model to optimize Neural Radiance Fields (NeRF) [25], achieving remarkable text-to-3D synthesis results. However, the method has two inherent limitations: (a) extremely slow optimization of NeRF and (b) low-resolution image space supervision on NeRF, leading to low-quality 3D models with a long processing time. In this paper, we address these limitations by utilizing a two-stage optimization framework. First, we obtain a coarse model using a low-resolution diffusion prior and accelerate with a sparse 3D hash grid structure. Using the coarse representation as the initialization, we further optimize a textured 3D mesh model with an efficient differentiable renderer interacting with a high-resolution latent diffusion model. Our method, dubbed `Magic3D`*, can create high quality 3D mesh models in 40 minutes, which is* $2\times$ *faster than DreamFusion (reportedly taking 1.5 hours on average), while also achieving higher resolution. User studies show 61.7% raters to prefer our approach over DreamFusion. Together with the image-conditioned generation capabilities, we provide users with new ways to control 3D synthesis, opening up new avenues to various creative applications. Project website:* `https://deepimagination.cc/Magic3D`

1. Introduction

3D digital content has been in high demand for a variety of applications, including gaming, entertainment, architecture, and robotics simulation. It is slowly finding its way into virtually every possible domain: retail, online conferencing, virtual social presence, education, *etc*. However, creating professional 3D content is not for anyone — it requires immense artistic and aesthetic training with 3D modeling expertise. Developing these skill sets takes a significant amount of time and effort. Augmenting 3D content creation with natural language could considerably help democratize 3D content creation for novices and turbocharge expert artists.

*†: equal contribution.

Image content creation from text prompts [2, 30, 35, 38] has seen significant progress with the advances of diffusion models [13, 43, 44] for generative modeling of images. The key enablers are large-scale datasets comprising billions of samples (images with text) scrapped from the Internet and massive amounts of compute. In contrast, 3D content generation has progressed at a much slower pace. Existing 3D object generation models [4, 9, 49] are mostly categorical. A trained model can only be used to synthesize objects for a single class, with early signs of scaling to multiple classes shown recently by Zeng *et al*. [49]. Therefore, what a user can do with these models is extremely limited and not yet ready for artistic creation. This limitation is largely due to the lack of diverse large-scale 3D datasets — compared to image and video content, 3D content is much less accessible on the Internet. This naturally raises the question of whether 3D generation capability can be achieved by leveraging powerful text-to-image generative models.

Recently, DreamFusion [33] demonstrated its remarkable ability for text-conditioned 3D content generation by utilizing a pre-trained text-to-image diffusion model [38] that generates images as a strong image prior. The diffusion model acts as a critic to optimize the underlying 3D representation. The optimization process ensures that rendered images from a 3D model, represented by Neural Radiance Fields (NeRF) [25], match the distribution of photorealistic images across different viewpoints, given the input text prompt. Since the supervision signal in DreamFusion operates on very low-resolution images (64×64), DreamFusion cannot synthesize high-frequency 3D geometric and texture details. Due to the use of inefficient MLP architectures for the NeRF representation, practical high-resolution synthesis may not even be possible as the required memory footprint and the computation budget grows quickly with the resolution. Even at a resolution of 64×64, optimization times are in hours (1.5 hours per prompt on average using TPUv4).

In this paper, we present a method that can synthesize highly detailed 3D models from text prompts within a reduced computation time. Specifically, we propose a coarse-to-fine optimization approach that uses multiple diffusion priors at different resolutions to optimize the 3D representa-

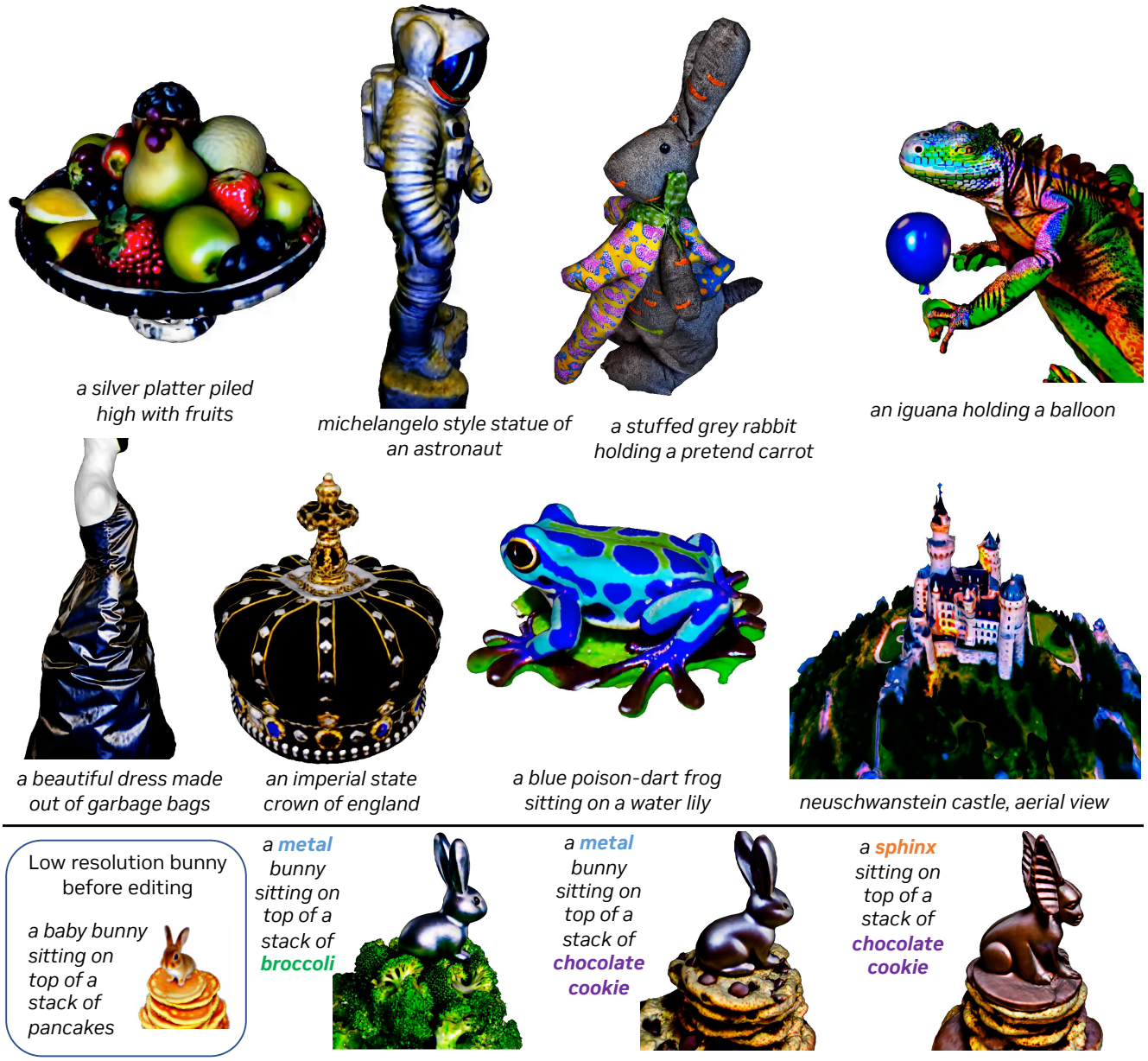


Figure 1. **Results and applications of Magic3D. Top: high-resolution text-to-3D generation**. Magic3D can generate high-quality and high-resolution 3D models from text prompts. **Bottom: high-resolution prompt-based editing**. Magic3D can edit 3D models by fine-tuning with the diffusion prior using a different prompt. Taking the low-resolution 3D model as the input (left), Magic3D can modify different parts of the 3D model corresponding to different input text prompts. Together with various creative controls on the generated 3D models, Magic3D is a convenient tool for augmenting 3D content creation.

tion, enabling the generation of both view-consistent geometry as well as high-resolution details. In the first stage, we optimize a coarse neural field representation akin to DreamFusion, but with a memory- and compute-efficient scene representation based on a hash grid [27]. In the second stage, we switch to optimizing mesh representations, a critical step that allows us to utilize diffusion priors at resolutions as high as 512×512. As 3D meshes are amenable to fast graphics renderers that can render high-resolution images in real-time, we leverage an efficient differentiable rasterizer [9, 28] and make use of camera close-ups to recover high-frequency details in geometry and texture. As a result, our approach produces high-fidelity 3D content (see Fig. 1) that can conveniently be imported and visualized in standard graphics

software and does so at 2× the speed of DreamFusion. Furthermore, we showcase various creative controls over the 3D synthesis process by leveraging the advancements developed for text-to-image editing applications [2, 37]. Our approach, dubbed Magic3D, endows users with unprecedented control in crafting their desired 3D objects with text prompts and reference images, bringing this technology one step closer to democratizing 3D content creation.

In summary, our work makes the following contributions:

- We propose Magic3D, a framework for high-quality 3D content synthesis using text prompts by improving several major design choices made in DreamFusion. It consists of a coarse-to-fine strategy that leverages both low- and high-resolution diffusion priors for learning the 3D representation of the target content. Magic3D, which synthesizes 3D content with an 8× higher resolution supervision, is also 2× faster than DreamFusion. 3D content synthesized by our approach is significantly preferable by users (61.7%).
- We extend various image editing techniques developed for text-to-image models to 3D object editing and show their applications in the proposed framework.

2. Related Work

Text-to-image generation. We have witnessed significant progress in text-to-image generation with diffusion models in recent years. With improvements in modeling and data curation, diffusion models can compose complex semantic concepts from text descriptions (nouns, adjectives, artistic styles, *etc*.) to generate high-quality images of objects and scenes [2, 35, 36, 38]. Sampling images from diffusion models is time consuming. To generate high-resolution images, these models either utilize a cascade of super-resolution models [2, 38] or sample from a lower-resolution latent space and decode latents into high-resolution images [36]. Despite the advances in high-resolution image generation, using language to describe and control 3D properties (*e.g*. camera viewpoints) while maintaining coherency in 3D remains an open, challenging problem.

3D generative models. There is a large body of work on 3D generative modeling, exploring different types of 3D representations such as 3D voxel grids [7, 12, 22, 42, 47], point-clouds [1, 23, 26, 48, 49, 51], meshes [9, 50], implicit [6, 24], or octree [15] representations. Most of these approaches rely on training data in the form of 3D assets, which are hard to acquire at scale. Inspired by the success of neural volume rendering [25], recent works started investing in 3D-aware image synthesis [4, 5, 10, 11, 29, 31, 32, 40], which has the advantage of learning 3D generative models directly from images — a more widely accessible resource. However, volume rendering networks are typically slow to query, leading to a trade-off between long training time [5, 31] and lack of multi-view consistency [10]. EG3D [4] partially mitigates this problem by utilizing a dual discriminator. While obtaining promising results, these works remain limited to modeling objects within a single object category, such as cars, chairs, or human faces, thus lacking scalability and the creative control desired for 3D content creation. In our paper, we focus on text-to-3D synthesis, aiming to generate a 3D renderable representation of a scene based on a text prompt.

Text-to-3D generation. With the recent success in text-to-image generative modeling in recent years, text-to-3D generation has also gained a surge of interest from the learning community. Earlier works such as CLIP-forge [39] synthesizes objects by learning a normalizing flow model to sample shape embeddings from textual input. However, it requires 3D assets in voxel representations during training, making it challenging to scale with data. DreamField [16] and CLIP-mesh [17] mitigate the training data issue by relying on a pre-trained image-text model [34] to optimize the underlying 3D representations (NeRFs and meshes), such that all 2D renderings reach high text-image alignment scores. While these approaches avoid the requirement of expensive 3D training data and mostly rely on pre-trained large-scale image-text models, they tend to produce less realistic 2D renderings.

Recently, DreamFusion [33] showcased impressive capability in text-to-3D synthesis by utilizing a powerful pre-trained text-to-image diffusion model [38] as a strong image prior. We build upon this work and improve over several design choices to bring significantly higher-fidelity 3D models into hands of users with a much reduced generation time.

3. Background: DreamFusion

DreamFusion [33] achieves text-to-3D generation with two key components: a neural scene representation which we refer to as the scene model, and a pre-trained text-to-image diffusion-based generative model. The scene model is a parametric function $x = g(\theta)$, which can produce an image x at the desired camera pose. Here, g is a volumetric renderer of choice, and θ is a coordinate-based MLP representing a 3D volume. The diffusion model ϕ comes with a learned denoising function $\epsilon_\phi(x_t; y, t)$ that predicts the sampled noise ϵ given the noisy image x_t, noise level t, and text embedding y. It provides the gradient direction to update θ such that all rendered images are pushed to the high probability density regions conditioned on the text embedding under the diffusion prior. Specifically, DreamFusion introduces Score Distillation Sampling (SDS), which computes the gradient:

$$\nabla_\theta \mathcal{L}_{\text{SDS}}(\phi, g(\theta)) = \mathbb{E}_{t,\epsilon}\left[w(t)(\epsilon_\phi(x_t; y, t) - \epsilon)\frac{\partial x}{\partial \theta}\right]. \quad (1)$$

Here, $w(t)$ is a weighting function. We view the scene model g and the diffusion model ϕ as modular components of the framework, amenable to choice. In practice, the denoising

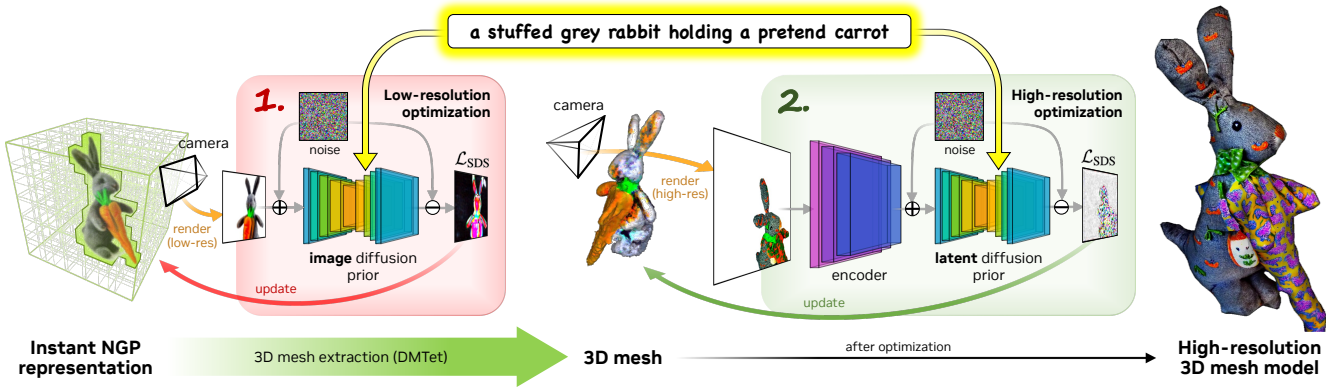


Figure 2. **Overview of `Magic3D`.** We generate high-resolution 3D content from an input text prompt in a coarse-to-fine manner. In the first stage, we utilize a low-resolution diffusion prior and optimize neural field representations (color, density, and normal fields) to obtain the coarse model. We further differentiably extract textured 3D mesh from the density and color fields of the coarse model. Then we fine-tune it using a high-resolution latent diffusion model. After optimization, our model generates high-quality 3D meshes with detailed textures.

function ϵ_ϕ is often replaced with another function $\tilde{\epsilon}_\phi$ that uses classifier-free guidance [14], which allows one to carefully weigh the strength of the text conditioning (see Sec. 6). DreamFusion relies on large classifier-free guidance weights to obtain results with better quality.

DreamFusion adopts a variant of Mip-NeRF 360 [3] with an explicit shading model for the scene model and Imagen [38] as the diffusion model. These choices result in two key limitations. First, high-resolution geometry or textures cannot be obtained since the diffusion model only operates on 64×64 images. Second, the utility of a large global MLP for volume rendering is both computationally expensive as well as memory intensive, making this approach scale poorly with the increasing resolution of images.

4. High-Resolution 3D Generation

`Magic3D` is a two-stage coarse-to-fine framework that uses efficient scene models that enable high-resolution text-to-3D synthesis (Fig. 2). We describe our method and key differences from DreamFusion [33] in this section.

4.1. Coarse-to-fine Diffusion Priors

`Magic3D` uses two different diffusion priors in a coarse-to-fine fashion to generate high-resolution geometry and textures. In the first stage, we use the base diffusion model described in eDiff-I [2], which is similar to the base diffusion model of Imagen [38] used in DreamFusion. This diffusion prior is used to compute gradients of the scene model via a loss defined on rendered images at a low resolution 64×64. In the second stage, we use the latent diffusion model (LDM) [36] that allows backpropagating gradients into rendered images at a high resolution 512×512; in practice, we choose to use the publicly available *Stable Diffusion* model [36]. Despite generating high-resolution images, the computation of LDM is manageable because the diffusion prior acts on the latent z_t with resolution 64×64:

$$\nabla_\theta \mathcal{L}_{\text{SDS}}(\phi, g(\theta)) = \mathbb{E}_{t,\epsilon}\left[w(t)(\epsilon_\phi(z_t; y, t) - \epsilon) \frac{\partial z}{\partial x} \frac{\partial x}{\partial \theta} \right]. \quad (2)$$

The increase in computation time mainly comes from computing $\partial x / \partial \theta$ (the gradient of the high-resolution rendered image) and $\partial z / \partial x$ (the gradient of the encoder in LDM).

4.2. Scene Models

We cater two different 3D scene representations to the two different diffusion priors at coarse and fine resolutions to accommodate the increased resolution of rendered images for the input of high-resolution priors, discussed as follows.

Neural fields as coarse scene models. The initial coarse stage of the optimization requires finding the geometry and textures from scratch. This can be challenging as we need to accommodate complex topological changes in the 3D geometry and depth ambiguities from the 2D supervision signals. In DreamFusion [33], the scene model is a neural field (a coordinate-based MLP) based on Mip-NeRF 360 [3] that predicts albedo and density. This is a suitable choice as neural fields can handle topological changes in a smooth, continuous fashion. However, Mip-NeRF 360 [3] is computationally expensive as it is based on a large global coordinate-based MLP. As volume rendering requires dense samples along a ray to accurately render high-frequency geometry and shading, the cost of having to evaluate a large neural network at every sample point quickly stacks up.

For this reason, we opt to use the hash grid encoding from Instant NGP [27], which allows us to represent high-frequency details at a much lower computational cost. We use the hash grid with two single-layer neural networks, one predicting albedo and density and the other predicting

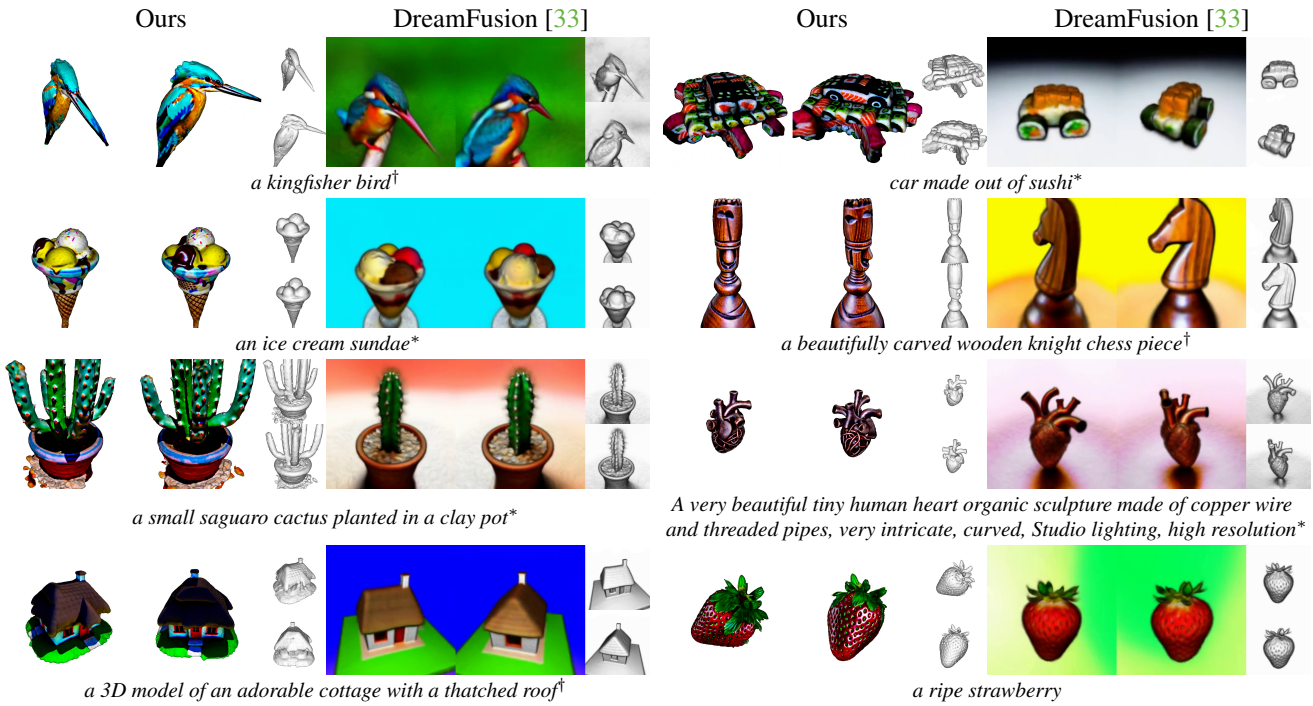


Figure 3. **Qualitative comparison with DreamFusion [33].** We use the same text prompt as in DreamFusion. For each 3D model, we render it from two views with a textureless rendering for each view and remove the background to focus on the actual 3D shape. For the DreamFusion results, we take frames from the videos published on the official webpage. Our `Magic3D` generates much higher quality 3D shapes on both geometry and texture compared with DreamFusion. $*$ *a DSLR photo of...* † *a zoomed out DSLR photo of...*

normals. We additionally maintain a spatial data structure that encodes scene occupancy and utilizes empty space skipping [20, 45]. Specifically, we use the density-based voxel pruning approach from Instant NGP [27] with an octree-based ray sampling and rendering algorithm [46]. With these design choices, we drastically accelerate the optimization of coarse scene models while maintaining quality.

Textured meshes as fine scene models. In our fine stage of optimization, we need to be able to accommodate very high-resolution rendered images to fine-tune our scene model with high-resolution diffusion priors. Using the same scene representation (the neural field) from the initial coarse stage of optimization could be a natural choice since the weights of the model can directly carry over. Although this strategy can work to some extent (Figs. 4 and 5), it struggles to render very high-resolution (*e.g.*, 512×512) images within reasonable memory constraints and computation budgets.

To resolve this issue, we use textured 3D meshes as the scene representation for the fine stage of optimization. In contrast to volume rendering for neural fields, rendering textured meshes with differentiable rasterization can be performed efficiently at very high resolutions, making meshes a suitable choice for our high-resolution optimization stage. Using the neural field from the coarse stage as the initialization for the mesh geometry, we can also sidestep the difficulty of learning large topological changes in meshes.

Formally, we represent the 3D shape using a deformable tetrahedral grid (V_T, T), where V_T is the vertices in the grid T [8, 41]. Each vertex $\mathbf{v}_i \in V_T \subset \mathbb{R}^3$ contains a signed distance field (SDF) value $s_i \in \mathbb{R}$ and a deformation $\Delta\mathbf{v}_i \in \mathbb{R}^3$ of the vertex from its initial canonical coordinate. Then, we extract a surface mesh from the SDF using a differentiable marching tetrahedra algorithm [41]. For textures, we use the neural color field as a volumetric texture representation.

4.3. Coarse-to-fine Optimization

We describe our coarse-to-fine optimization procedure, which first operates on a coarse neural field representation and subsequently a high-resolution textured mesh.

Neural field optimization. Similarly to Instant NGP [27], we initialize an occupancy grid of resolution 256^3 with values to 20 to encourage shapes to grow in the early stages of optimization. We update the grid every 10 iterations and generate an octree for empty space skipping. We decay the occupancy grid by 0.6 in every update and follow Instant NGP with the same update and thresholding parameters.

Instead of estimating normals from density differences, we use an MLP to predict the normals. Note that this does not violate geometric properties since volume rendering is used instead of surface rendering; as such, the orientation

Table 1. **User preference study.** We conducted a user study to measure preference for 3D models generated using 397 prompts released by DreamFusion. Significantly more raters (61.7%) prefer 3D models generated by `Magic3D` over DreamFusion. The majority of raters (87.7%) prefer fine models over coarse models in `Magic3D`, showing the effectiveness of our coarse-to-fine approach.

Comparison	*Preference (%)*
Ours vs. DreamFusion [33]	61.7
Ours vs. Ours (coarse)	87.7

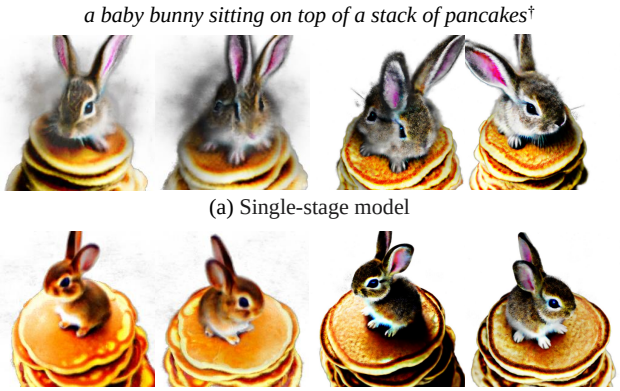
of particles at continuous positions need not be oriented to the level set surface. This helps us significantly reduce the computational cost of optimizing the coarse model by avoiding the use of finite differencing. Accurate normals can be obtained in the fine stage of optimization when we use a true surface rendering model.

Similar to DreamFusion, we also model the background using an environment map MLP, which predicts RGB colors as a function of ray directions. Since our sparse representation model does not support scene reparametrization as in Mip-NeRF 360 [3], the optimization has a tendency to "cheat" by learning the essence of the object using the background environment map. As such, we use a tiny MLP for the environment map (hidden dimension size of 16) and weigh down the learning rate by $10\times$ to allow the model to focus more on the neural field geometry.

Mesh optimization. To optimize a mesh from the neural field initialization, we convert the (coarse) density field to an SDF by subtracting it with a non-zero constant, yielding the initial s_i. We additionally initialize the volume texture field directly with the color field optimized from the coarse stage.

During optimization, we render the extracted surface mesh into high-resolution images using a differentiable rasterizer [19,28]. We optimize both s_i and $\Delta\mathbf{v}_i$ for each vertex $\mathbf{v}_i$ via backpropagation using the high-resolution SDS gradient (Eq. 2). When rendering the mesh to an image, we also track the 3D coordinates of each corresponding pixel projection, which would be used to query colors in the corresponding texture field for joint optimization.

When rendering the mesh, we increase the focal length to zoom in on object details, which is a critical step towards recovering high-frequency details. We keep the same pre-trained environment map from the coarse stage of optimization and composite the rendered background with the rendered foreground object using differentiable antialiasing [19]. To encourage the smoothness of the surface, we further regularize the angular differences between adjacent faces on the mesh. This allows us to obtain well-behaved geometry even under supervision signals with high variance, such as the SDS gradient $\nabla_\theta \mathcal{L}_{\text{SDS}}$.



(a) Single-stage model

(b) Coarse-to-fine model

Figure 4. **Single-stage (top) vs. coarse-to-fine models (bottom).** Both use NeRF for the scene model. The left two columns use 64×64 rendering resolution during optimization while the right two use 256×256. Compared to our coarse-to-fine approach, the single-stage method can generate details but with worse shapes.

5. Experiments

We focus on comparing our method with DreamFusion [33] on 397 text prompts taken from the website of DreamFusion[1]. We train `Magic3D` on all of the text prompts and compare them with the results provided on the website.

Speed evaluation. Unless otherwise noted, the coarse stage is trained for 5000 iterations with 1024 samples along the ray (subsequently filtered by the sparse octree) with a batch size of 32, with a total runtime of around 15 minutes (upwards of 8 iterations / second, variable due to differences in sparsity). The fine stage is trained for 3000 iterations with a batch size of 32 with a total runtime of 25 minutes (2 iterations / second). Both runtimes combined are 40 minutes. All runtimes were measured on 8 NVIDIA A100 GPUs.

Qualitative comparisons. We provide qualitative examples in Fig. 3. Qualitatively, our models achieve much higher 3D quality in terms of both geometry and texture. Notice that our model can generate candies on ice cream cones, highly detailed sushi-like cars, vivid strawberries, and birds. We also note that our resulting 3D models can be directly imported and visualized in standard graphics software.

User studies. We conduct user studies to evaluate different methods based on user preferences on Amazon MTurk. We show users two videos side by side rendered from a canonical view by two different algorithms using the same text prompt. We ask the users to select the one that is more realistic and detailed. Each prompt is evaluated by 3 different users, resulting in 1191 pairwise comparisons. As shown in Table 1, users favor 3D models generated by `Magic3D`, with 61.7% of the users considering our results with higher quality.

[1] https://dreamfusion3d.github.io/gallery.html

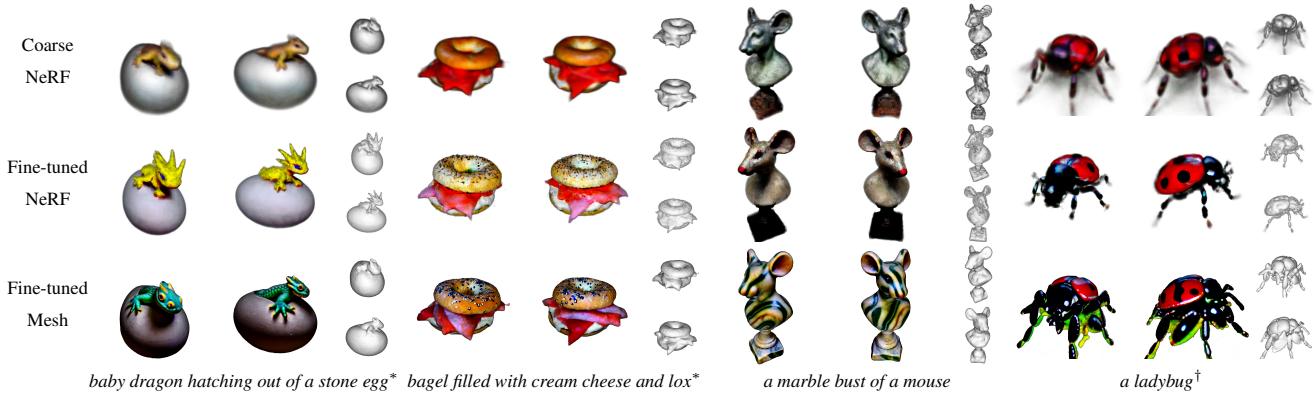


Figure 5. **Ablation on the fine-tuning stage.** For each text prompt, we compare coarse and fine models with mesh and NeRF representations. Mesh fine-tuning significantly improve the visual quality of generated 3D assets, providing more photo-realistic details on the 3D shapes.

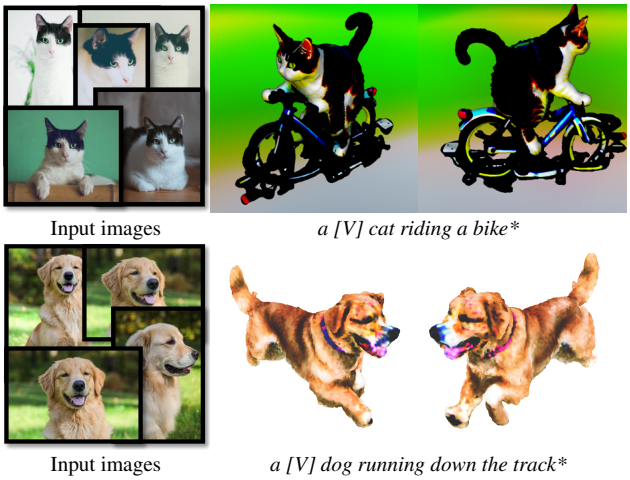


Figure 6. `Magic3D` **with DreamBooth-based personalization.** Given an input image of a particular instance, we fine-tune the diffusion models with DreamBooth and optimize the 3D models with the given prompts. The identity is well preserved in the generated 3D models. *Image source (input images): Unsplash.*

Can single-stage optimization work with LDM prior? We ablate scene models optimized with high-resolution LDM prior in a single-stage optimization setup. We find that 3D meshes as the scene model fail to generate high-quality results if optimized from scratch. This leaves our our memory-efficient sparse 3D representation as the ideal candidate for the scene model. However, rendering 512×512 images is still too memory intensive to fit into modern GPUs. Therefore, we render lower-resolution images from the scene model and upsample them to 512×512 as input to the LDM. We find it generates objects with worse shapes. Fig. 4 shows two examples with scene rendering resolution 64×64 and 256×256 respectively (top row). While it generates furry details, the shape is worse than the coarse model.

Can we use NeRF for the fine model? Yes. While optimizing a NeRF from scratch does not work well, we can follow the coarse-to-fine framework but replace the second-stage scene model with a NeRF. In the bottom right of Fig. 4, we show the result of a fine NeRF model initialized with the coarse model on its left and fine-tuned with 256×256 rendered images. The two-stage approach retains good geometry in the initial model and adds more details, showing superior quality to its one-stage counterpart.

Coarse models vs. fine models. Fig. 5 provides more visual results contrasting coarse and fine models. We try both NeRF and mesh for scene models and fine-tune them from the same coarse model above. We see significant quality improvements on both NeRF and mesh models, suggesting our coarse-to-fine approach works for general scene models.

6. Controllable 3D Generation

As certain styles and concepts are difficult to express in words but easy with images, it is desirable to have a mechanism to influence the text-to-3D model generation with images. We explore different image conditioning techniques as well as a prompt-based editing approach to provide users more control over the 3D generation outputs.

Personalized text-to-3D. DreamBooth [37] described a method to personalize text-to-image diffusion models by fine-tuning a pre-trained model on several images of a subject. The fine-tuned model can learn to tie the subject to a unique identifier string, denoted as `[V]`, and generate images of the subject when `[V]` is included in the text prompt. In the context of text-to-3D generation, we would like to generate a 3D model of a subject. This can be achieved by first fine-tuning our diffusion prior models with the DreamBooth approach, and then using the fine-tuned diffusion priors with the `[V]` identifier as part of the conditioning text prompt to provide the learning signal when optimizing the 3D model.

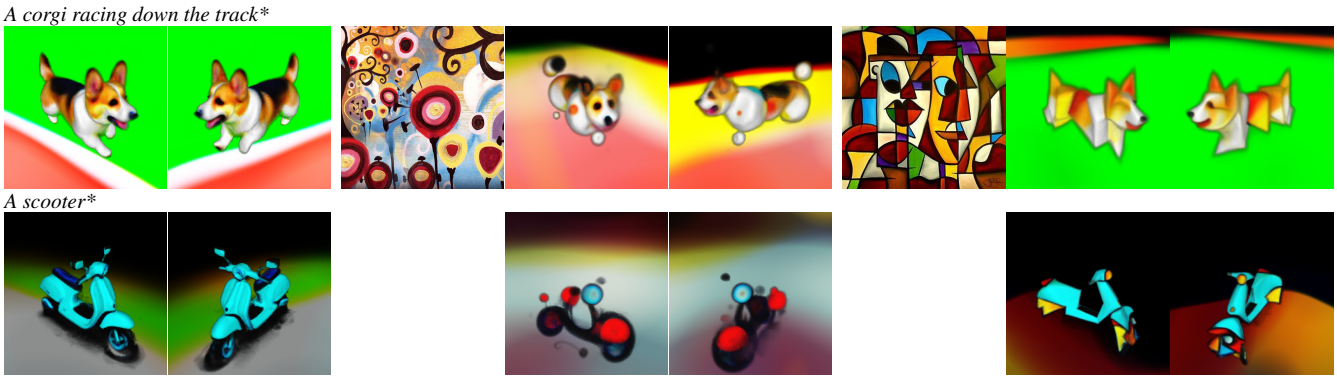


Figure 7. `Magic3D` **with image style transfer.** The style of the reference image is transferred to the 3D model by providing it to the diffusion model as conditional input. We apply different styles during 3D synthesis with two different text prompts. The first two columns show the 3D model optimized given the text without reference image. Afterwards, we show the reference and the corresponding 3D shapes.

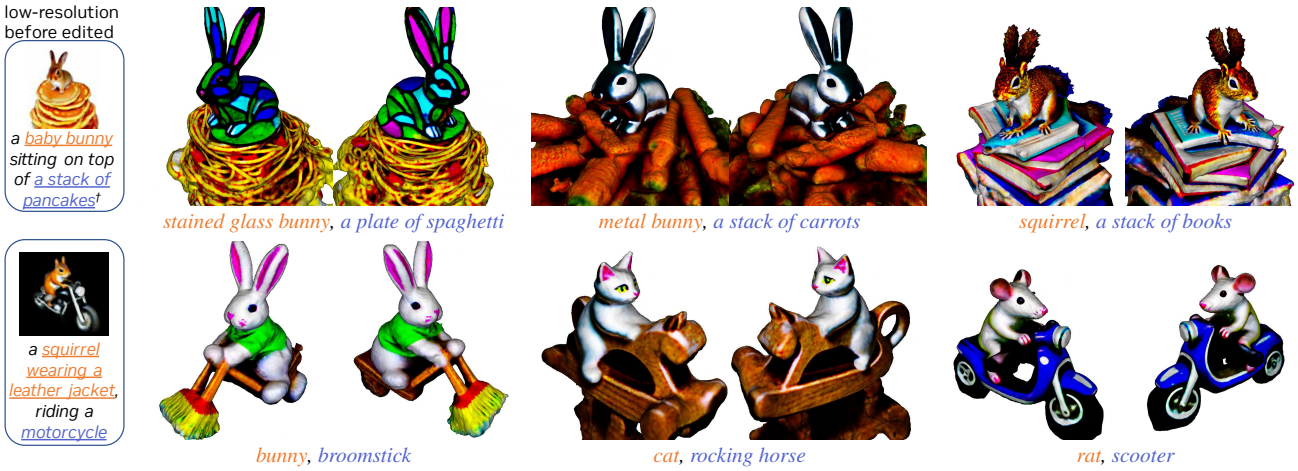


Figure 8. `Magic3D` **with prompt-based editing**. Given a coarse model (first column) generated with a base prompt, we replace the underscored text with new text and fine-tune the NeRF to get a high-resolution NeRF model with LDM. We further fine-tune the high-resolution mesh with the NeRF model. Such a prompt-based editing method gives artists better control over the 3D generation output.

To demonstrate the applicability of DreamBooth in our framework, we collect 11 images of one cat and 4 images of one dog. We fine-tune eDiff-I [2] and LDM [36], binding the text identifier `[V]` to the given subject. Then, we optimize the 3D model with `[V]` in the text prompts. We use a batch size of 1 for all fine-tuning. For eDiff-I, we use the Adam optimizer with learning rate 1×10^{-5} for 1,500 iterations; for LDM, we fine-tune with learning rate 1×10^{-6} for 800 iterations. Fig. 6 shows our personalized text-to-3D results: we are able to successfully modify the 3D models preserving the subjects in the given input images.

Style-guided text-to-3D. We also explore controlling the 3D generation with multi-modal conditioning. The eDiff-I diffusion prior [2] is designed such that it can condition on a reference image when performing text-to-image generation. Such an image conditioning design makes it easy to change the style of the generated output. However, we find that naïvely feeding the style image as input to the model when computing the SDS gradient can result in a poor 3D model that is essentially overfitting to the input image. We hypothesize that the conditioning signal by the image is significantly stronger than the text prompt during optimization. To better balance the guidance strength between image and text conditioning, we extend our model's classifier-free guidance scheme [14] and compute the final $\tilde{\epsilon}_\phi(x_t; y_{\text{text}}, y_{\text{image}}, t)$:

$$\begin{aligned}\tilde{\epsilon}_\phi(x_t; y_{\text{text}}, y_{\text{image}}, t) &= \epsilon_\phi(x_t; t) \\ &+ \omega_{\text{text}}[\epsilon_\phi(x_t; y_{\text{text}}, t) - \epsilon_\phi(x_t; t)] \\ &+ \omega_{\text{joint}}[\epsilon_\phi(x_t; y_{\text{text}}, y_{\text{image}}, t) - \epsilon_\phi(x_t; t)] \,, \end{aligned} \tag{3}$$

where y_{text} and y_{image} are text and image conditioning respectively, and ω_{text} and ω_{joint} are the guidance weights for text and joint text-and-image conditioning respectively. Note that for $\omega_{\text{joint}} = 0$, the scheme is equivalent to standard classifier-free guidance with respect to text conditioning only.

Fig. 7 shows our style-guided text-to-3D generation results. When optimizing the 3D model, we feed the reference image to the eDiff-I model. We set $\omega_{\text{text}}, \omega_{\text{joint}} = 50, 50$ or $40, 60$ and apply the image guidance when $t < 0.5$ only. We do not provide high-resolution results for this experiment because LDM does not support reference image conditioning.

Prompt-based editing through fine-tuning. Another way to control the generated 3D content is by fine-tuning a learned coarse model with a new prompt. Our prompt-based editing includes three stages. (a) We train a coarse model with a base prompt. (b) We modify the base prompt and fine-tune the coarse model with the LDM. This stage provides a well initialized NeRF model for the next step. Directly applying mesh optimization on a new prompt would generate highly detailed textures but could deform geometry only slightly. (c) We optimize the mesh with the modified text prompt. Our prompt-based editing can modify the texture of the shape or transform the geometry and texture according to the text. The resulting scene models preserve the layer-out and overall structure. Such an editing capability makes the 3D content creation with `Magic3D` more controllable. In Fig. 8, we show two coarse NeRF models trained with the base prompt for the "bunny" and "squirrel". We modify the base prompt, fine-tune the NeRF model in high resolution and optimize the mesh. Results show that we can tune the scene model according to the prompt, *e.g.* changing the "baby bunny" to "stained glass bunny" or "metal bunny" results in similar geometry but with a different texture.

7. Conclusion

We propose `Magic3D`, a fast and high-quality text-to-3D generation framework. We benefit from both efficient scene models and high-resolution diffusion priors in a coarse-to-fine approach. In particular, the 3D mesh models scale nicely with image resolution and enjoy the benefits of higher resolution supervision brought by the latent diffusion model without sacrificing its speed. It takes 40 minutes from a text prompt to a high-quality 3D mesh model ready to be used in graphic engines. With extensive user studies and qualitative comparisons, we show that `Magic3D` is more preferable (61.7%) by the raters compared to DreamFusion, while enjoying a $2\times$ speed-up. Lastly, we propose a set of tools for better controlling style and content in 3D generation. We hope with `Magic3D`, we can democratize 3D synthesis and open up everyone's creativity in 3D content creation.

Acknowledgements. We would like to thank Frank Shen, Yogesh Balaji, Seungjun Nah, James Lucas, David Luebke, Clement Fuji-Tsang, Charles Loop, Qinsheng Zhang, Zan Gojcic, and Jonathan Tremblay for helpful discussions and paper proofreading. We would also like to thank Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall for providing additional implementation details in DreamFusion.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pages 40–49. PMLR, 2018. 3

[2] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, et al. ediff-i: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022. 1, 3, 4, 8, 12

[3] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022. 4, 6, 11

[4] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16123–16133, 2022. 1, 3

[5] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5799–5809, 2021. 3

[6] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019. 3

[7] Matheus Gadelha, Subhransu Maji, and Rui Wang. 3d shape induction from 2d views of multiple objects. In *2017 International Conference on 3D Vision (3DV)*, pages 402–411. IEEE, 2017. 3

[8] Jun Gao, Wenzheng Chen, Tommy Xiang, Clement Fuji Tsang, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Learning deformable tetrahedral meshes for 3d reconstruction. In *Advances In Neural Information Processing Systems*, 2020. 5

[9] 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. *Advances in Neural Information Processing Systems*, 2022. 1, 2, 3

[10] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d aware generator for high-resolution image synthesis. In *International Conference on Learning Representations*, 2022. 3

[11] Zekun Hao, Arun Mallya, Serge Belongie, and Ming-Yu Liu. GANcraft: Unsupervised 3D Neural Rendering of Minecraft Worlds. In *ICCV*, 2021. 3

[12] Philipp Henzler, Niloy J. Mitra, and Tobias Ritschel. Escaping plato's cave: 3d shape from adversarial rendering. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 3

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 1

[14] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. 4, 8

[15] Moritz Ibing, Gregor Kobsik, and Leif Kobbelt. Octree transformer: Autoregressive 3d shape generation on hierarchically structured sequences. *arXiv preprint arXiv:2111.12480*, 2021. 3

[16] Ajay Jain, Ben Mildenhall, Jonathan T. Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. 2022. 3

[17] Nasir Khalid, Tianhao Xie, Eugene Belilovsky, and Tiberiu Popa. Clip-mesh: Generating textured meshes from text using pretrained image-text models. *ACM Transactions on Graphics (TOG), Proc. SIGGRAPH Asia*, 2022. 3

[18] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. 12

[19] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics*, 39(6), 2020. 6

[20] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020. 5

[21] Nan Liu, Shuang Li, Yilun Du, Antonio Torralba, and Joshua B Tenenbaum. Compositional visual generation with composable diffusion models. *arXiv preprint arXiv:2206.01714*, 2022. 12

[22] Sebastian Lunz, Yingzhen Li, Andrew Fitzgibbon, and Nate Kushman. Inverse graphics gan: Learning to generate 3d shapes from unstructured 2d data. *arXiv preprint arXiv:2002.12674*, 2020. 3

[23] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 3

[24] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019. 3

[25] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 3

[26] Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy Mitra, and Leonidas Guibas. Structurenet: Hierarchical graph networks for 3d shape generation. *ACM Transactions on Graphics (TOG), Siggraph Asia 2019*, 38(6):Article 242, 2019. 3

[27] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. 2, 4, 5, 11

[28] Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. Extracting triangular 3d models, materials, and lighting from images. In *CVPR*, pages 8280–8290, 2022. 2, 6

[29] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7588–7597, 2019. 3

[30] 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. 1

[31] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11453–11464, 2021. 3

[32] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. Stylesdf: High-resolution 3d-consistent image and geometry generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13503–13513, 2022. 3

[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, 4, 5, 6, 11, 12, 15, 16, 17, 18, 19

[34] 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. 3

[35] 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. 1, 3

[36] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, June 2022. 3, 4, 8

[37] 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. 3, 7

[38] 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, et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022. 1, 3, 4, 12

[39] Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshan. Clip-forge: Towards zero-shot text-to-shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18603–18613, 2022. 3

[40] Katja Schwarz, Axel Sauer, Michael Niemeyer, Yiyi Liao, and Andreas Geiger. Voxgraf: Fast 3d-aware image synthesis with sparse voxel grids. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 3

[41] 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. 5

[42] Edward J Smith and David Meger. Improved adversarial systems for 3d object generation and reconstruction. In *Conference on Robot Learning*, pages 87–96. PMLR, 2017. 3

[43] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015. 1

[44] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019. 1

[45] Towaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11358–11367, 2021. 5

[46] Towaki Takikawa, Or Perel, Clement Fuji Tsang, Charles Loop, Joey Litalien, Jonathan Tremblay, Sanja Fidler, and Maria Shugrina. Kaolin wisp: A pytorch library and engine for neural fields research. https://github.com/NVIDIAGameWorks/kaolin-wisp, 2022. 5

[47] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *Advances in neural information processing systems*, 29, 2016. 3

[48] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4541–4550, 2019. 3

[49] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 1, 3

[50] Yuxuan Zhang, Wenzheng Chen, Huan Ling, Jun Gao, Yinan Zhang, Antonio Torralba, and Sanja Fidler. Image gans meet differentiable rendering for inverse graphics and interpretable 3d neural rendering. In *International Conference on Learning Representations*, 2021. 3

[51] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5826–5835, 2021. 3

Appendix

A. Author Contributions

All authors have significant contributions on ideas, explorations, and paper writing. Specifically, **CHL** and **TYL** led the research, developed fundamental code for experiments and organized team efforts. **JG** led the experiments on generating high-resolution mesh models. **LT** led the experiments on using high-resolution diffusion prior. **TT** led the experiments on sparse scene representations. **XZ** and **KK** led the experiments in controllable generation. **XH** conducted the user study. **SF** and **MYL** advised the research direction and designed the scope of the project.

B. Implementation Details

We follow the implementation details described by Poole *et al*. [33] as closely as possible. We refer readers to the Dreamfusion paper [33] for context and list the major differences below.

Architectural details. As aforementioned in the main paper, we adopt a multi-resolution hash grid encoding architecture from Instant NGP [27] instead of using a large global coordinate-based MLP architecture. We use 16 levels of hash dictionaries of size 2^{19} and dimension 4, spanning 3D gird resolutions from 2^4 to 2^{12} with an exponential growth rate. We use single-layer MLPs with 32 hidden units to predict all of RGB color, volume density, and normal, where the inputs to the MLPs are the concatenated feature vectors from the multi-resolution hash encoding sampled with trilinear interpolation (we refer readers to the Instant NGP paper [27] for more details in this representation). We perform density-based pruning to sparsify the Instant NGP representation with an octree structure every 10 iterations. This allows us to more efficiently render pixels using empty space skipping, even with 3D points as dense as 1024 samples per ray. We do not use the contracting reparametrization of unbounded scenes from Mip-NeRF 360 [3] as it is not supported by our sparse representation.

Scene representation. For the coarse neural field representation, we use a bounding sphere of radius 2 for our experiments. We use the $\mathrm{softplus}$ activation for the density prediction and follow Poole *et al*. [33] to add an initial spatial density bias to encourage the optimization to focus on the object-centric neural field. We empirically found that using a linear form of spatial density bias helps stabilize the optimization, more formally written as

$$\tau_{\text{init}}(\boldsymbol{\mu}) = \lambda_\tau \cdot \left(1 - \frac{\|\boldsymbol{\mu}\|_2}{c}\right) , \tag{4}$$

where $\boldsymbol{\mu}$ is the 3D location, $\lambda_\tau = 10$ is the density bias scale, and $c = 0.5$ is an offset scale. Different from DreamFusion,

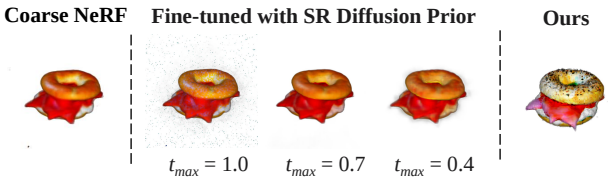


Figure 9. Fine-tuning NeRF with SR prior fails to add high-resolution details. t_{max} is the maximum timestep sampled in SDS.

however, we add this density bias to the *pre*-activation; as a result, the post-activation of the density prediction will vary continuously from $\text{softplus}(\lambda_\tau)$ to 0 as a function of $\|\boldsymbol{\mu}\|_2$.

Camera and light augmentations. We follow Poole *et al*. [33] to add random augmentations to the camera and light sampling for rendering the shaded images. Differently, (a) we sample the point light location such that the angular distance from the random camera center location (w.r.t. the origin) is sampled from $\psi_{\text{cam}} \sim \mathcal{U}(0, \pi/3)$ with a random point light distance $r_{\text{cam}} \sim \mathcal{U}(0.8, 1.5)$, and (b) we use a "soft" version of the textureless and albedo-only augmentation such that various strengths of shading in the rendered images are seen during optimization. (c) we sample the camera distance from $\mathcal{U}(1.5, 2)$, and the focal length $\mathcal{U}(0.7, 1.35)$. When training with high resolution diffusion prior, we increase the focal length and sample from $\mathcal{U}(1.2, 1.8)$.

Optimization. Unless otherwise specified, we optimize the coarse model with batch size 32 using the Adam optimizer [18] with a learning rate of 1×10^{-2} without warmup and decay. Note that the large global coordinate-based MLP architecture in DreamFusion [33] limits its optimization to only an effective batch size of 8. For the coarse model, we add the opacity regularization as suggested by Poole *et al*. [33] to encourage sparsity in the volume density field, but we do not add the orientation regularization as we empirically found it to hurt optimization.

Score Distillation Sampling. In the first stage, we sample the timestep $t \sim \mathcal{U}(0, 1)$ and set $w(t) = 1$. In the second stage, we find the range of timestep t in SDS affects the quality. We sample $t \sim \mathcal{U}(0.02, 0.5)$ in our experiments. In general, setting t_{max} in the range of $[0.5, 0.7]$ works well. We set $w(t) = \sigma_t^2$ in this stage.

C. Alternative High-Resolution Prior

In addition to LDM, we also consider using Super Resolution (SR) diffusion prior [2, 38] for increasing the resolution of a coarse model. This diffusion model is trained to generate a high-resolution image conditioning on a low-resolution input image. In SDS, this model predicts noises added in high resolution, i.e., $\epsilon_\phi(x_t; y, t, x_{\text{low}})$, where x_{low} denotes a 64×64 low-resolution image. We render x_{low} with a frozen coarse model to optimize the second-stage fine model. Fig. 9 shows this approach fails to add high-quality details to the input coarse model.

D. Ablation Studies

Style-guided text-to-3D: guidance weight and noise level threshold. We ablate different combinations of guidance weights and noise level thresholds in Figs. 10 and 11, respectively. The guidance weights ω_{text} and ω_{joint} balance the guidance strength during optimization (see Eq. 3). A similar guidance formulation has also been used by Liu *et al*. for compositional text-to-image generation [21]. We also find that applying the image conditioning only below a certain noise level threshold can help control style transfer. The intuition is that image-based style guidance is most relevant for optimizing the generated 3D object's details, which are modeled at lower noise levels. Notice that we do not provide high-resolution results for this experiment because LDM does not support image conditioning inputs.

Style-guided text-to-3D: content image as reference. We also explore using multiple images as inputs during 3D synthesis to transfer the content in the images to the 3D model, as shown in Fig. 12: Given a text prompt, we first ask the eDiff-I model to generate the front view, side view and back view images. When optimizing the 3D model for the same text prompt from different views, we then feed the corresponding generated view image as input to guide the 3D synthesis. This approach requires some degree of consistency with respect to subject identity across the different view images, which can be achieved by generating a set of different view images first and choosing accordingly. Overall, the experiment shows that we can apply the text-to-image diffusion model to generate images that can be used for guidance during 3D model optimization. As we see, this does not only provide enhanced control by preserving the identity of the subjects in the images, but also improves output quality and 3D consistency. Generally, depending on image type, image conditioning can be used either for object-centric content transfer to 3D (Fig. 12) or for abstract 3D stylization (Figs. 7, 10, and 11).

E. More Qualitative Results

We provide more qualitative comparisons with Dreamfusion [33] in Figs. 14, 15, 16, 17, 18. Our `Magic3D` achieved much higher quality in terms 3D geometry and texture.

We also show more results on prompt-based editing in Fig. 13. Our `Magic3D` enable high-quality editing of the 3D content through simple text prompt modification.

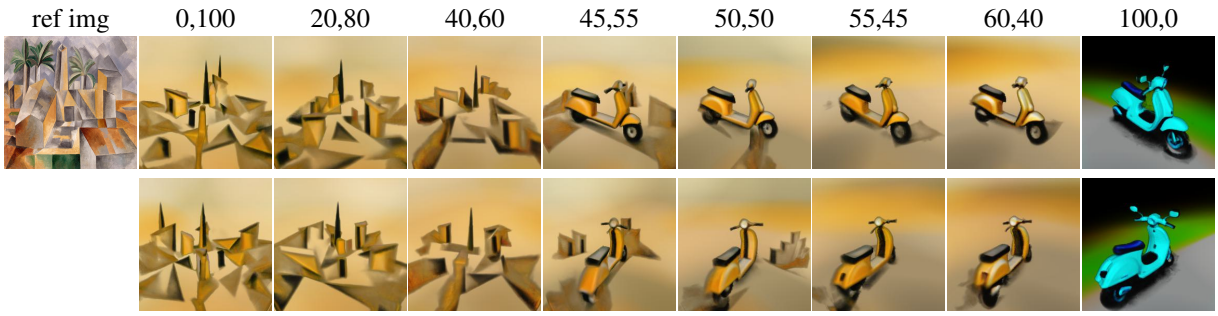


Figure 10. `Magic3D` **with image style transfer with different guidance weights.** Each column uses a different combination of guidance weights $\omega_{\text{text}}, \omega_{\text{joint}}$. All results are optimized with noise level threshold $t = 1.0$. When $\omega_{\text{text}} = 0, \omega_{\text{joint}} = 100$, the setting is equivalent to using a single guidance weight. The style image dominates the resulting scene if the ω_{joint} is too large. We generally find that using a guidance weight combination around $50, 50$ results in the best performance.

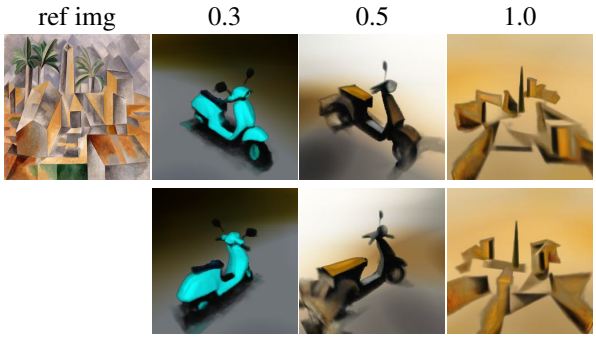


Figure 11. `Magic3D` **with image style transfer with different noise level thresholds.** Each column uses a different noise level threshold t. All results are optimized with guidance weights ω_{text}, $\omega_{\text{joint}} = 40, 60$. When the noise level threshold $t = 0$, the setup is equivalent to using no style image guidance. We generally find that setting the threshold around 0.5 provides the best performance.

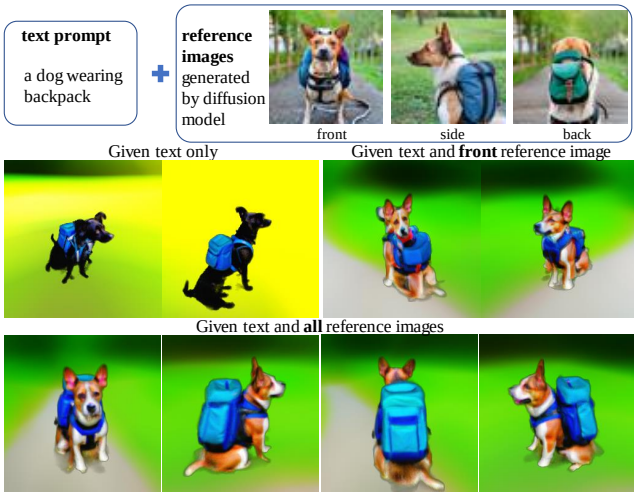


Figure 12. `Magic3D` **with image content transfer.** Given the reference images generated by the diffusion prior, we optimize 3D models that look similar to the object in the images. We show the generated 3D models given *(i)* text only, *(ii)* text and front view's reference image only and *(iii)* text and different view's reference images. *(ii)* and *(iii)* preserve the look of the dog in the image. With multiple reference images, *(iii)* yields higher quality and more 3D-consistent outputs.

Figure 13. `Magic3D` **with prompt-based editing**. Given a coarse model (first column) generated with a base prompt, we replace the underscored text with new text and fine-tune the NeRF to get a high-resolution NeRF model with LDM. We further fine-tune the high-resolution mesh on top of the NeRF model. Such a prompt-based editing technique gives artists better control over the 3D generation output.

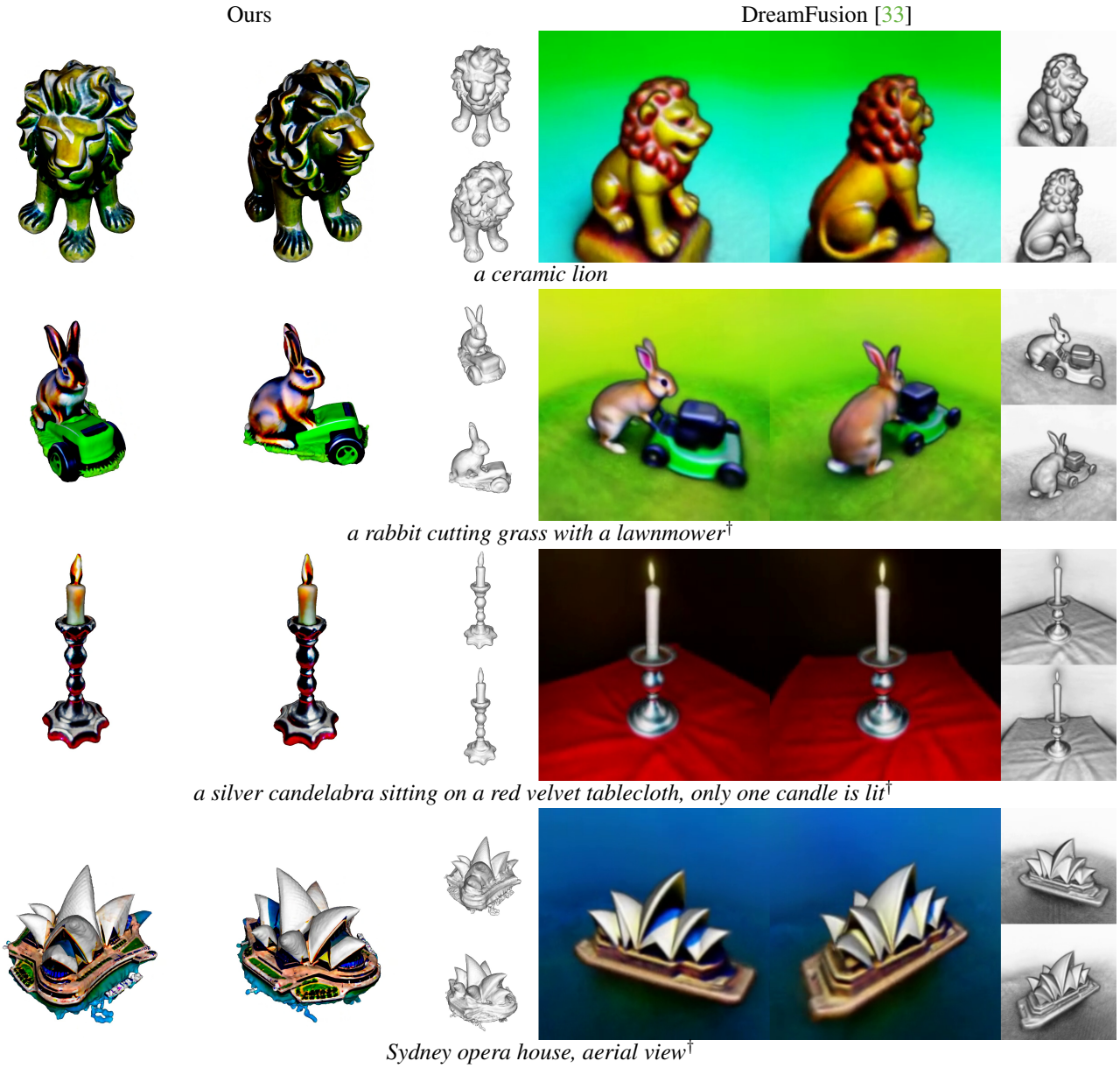


Figure 14. **Qualitative comparison with DreamFusion [33].** We use the same text prompt as in DreamFusion. For each 3D model, we render it from two views with a textureless rendering for each view and remove the background to focus on the 3D shape. For the DreamFusion results, we take frames from the videos published on the official webpage. `Magic3D` generates much higher quality 3D shapes on both geometry and texture compared with DreamFusion. $*$ *a DSLR photo of...* † *a zoomed out DSLR photo of...*

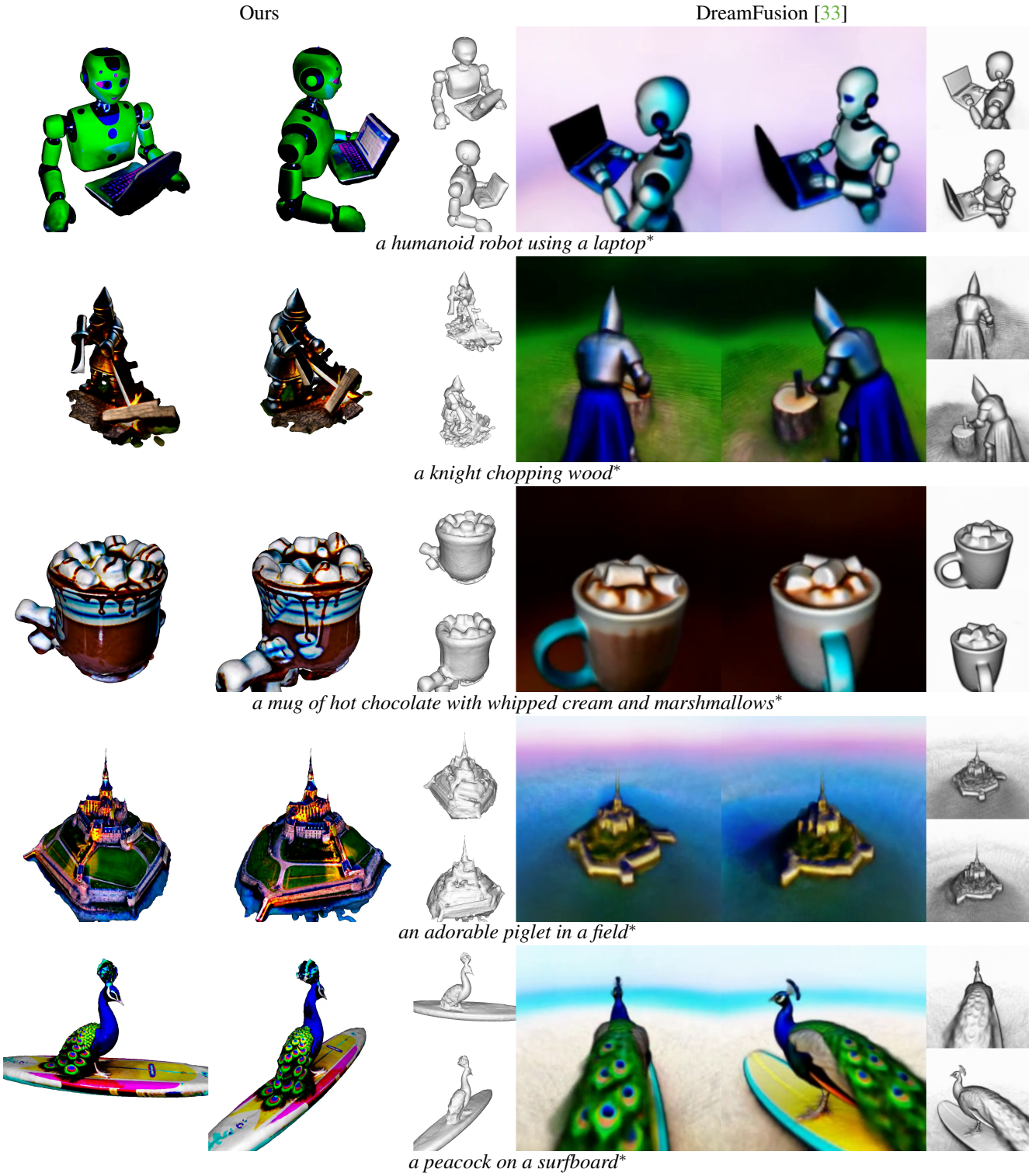


Figure 15. **Qualitative comparison with DreamFusion [33].** We use the same text prompt as in DreamFusion. For each 3D model, we render it from two views with a textureless rendering for each view and remove the background to focus on the 3D shape. For the DreamFusion results, we take frames from the videos published on the official webpage. `Magic3D` generates much higher quality 3D shapes on both geometry and texture compared with DreamFusion. $*$ *a DSLR photo of...* † *a zoomed out DSLR photo of...*

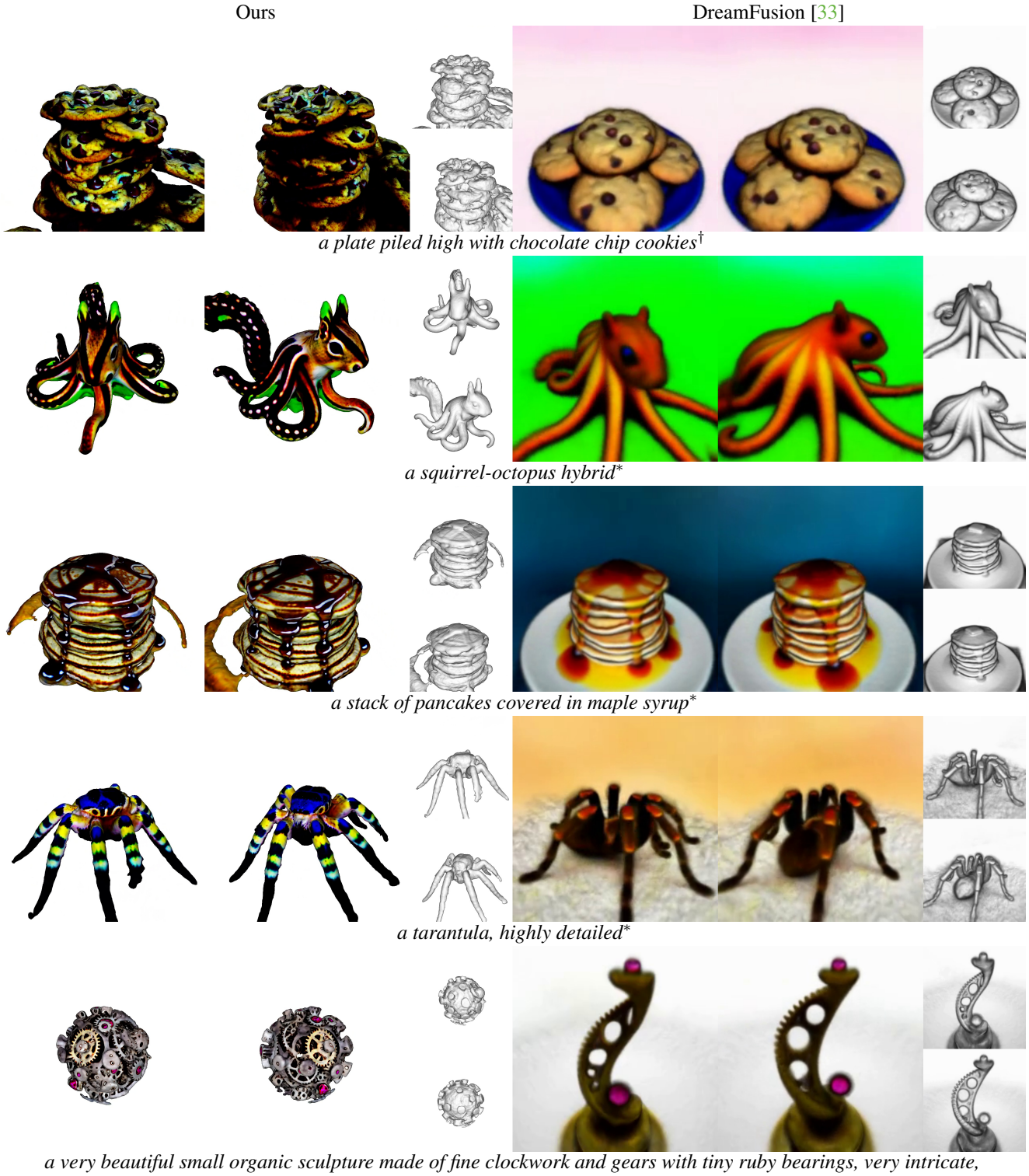


Figure 16. **Qualitative comparison with DreamFusion [33].** We use the same text prompt as in DreamFusion. For each 3D model, we render it from two views with a textureless rendering for each view and remove the background to focus on the 3D shape. For the DreamFusion results, we take frames from the videos published on the official webpage. `Magic3D` generates much higher quality 3D shapes on both geometry and texture compared with DreamFusion. ∗ *a DSLR photo of...* † *a zoomed out DSLR photo of...*

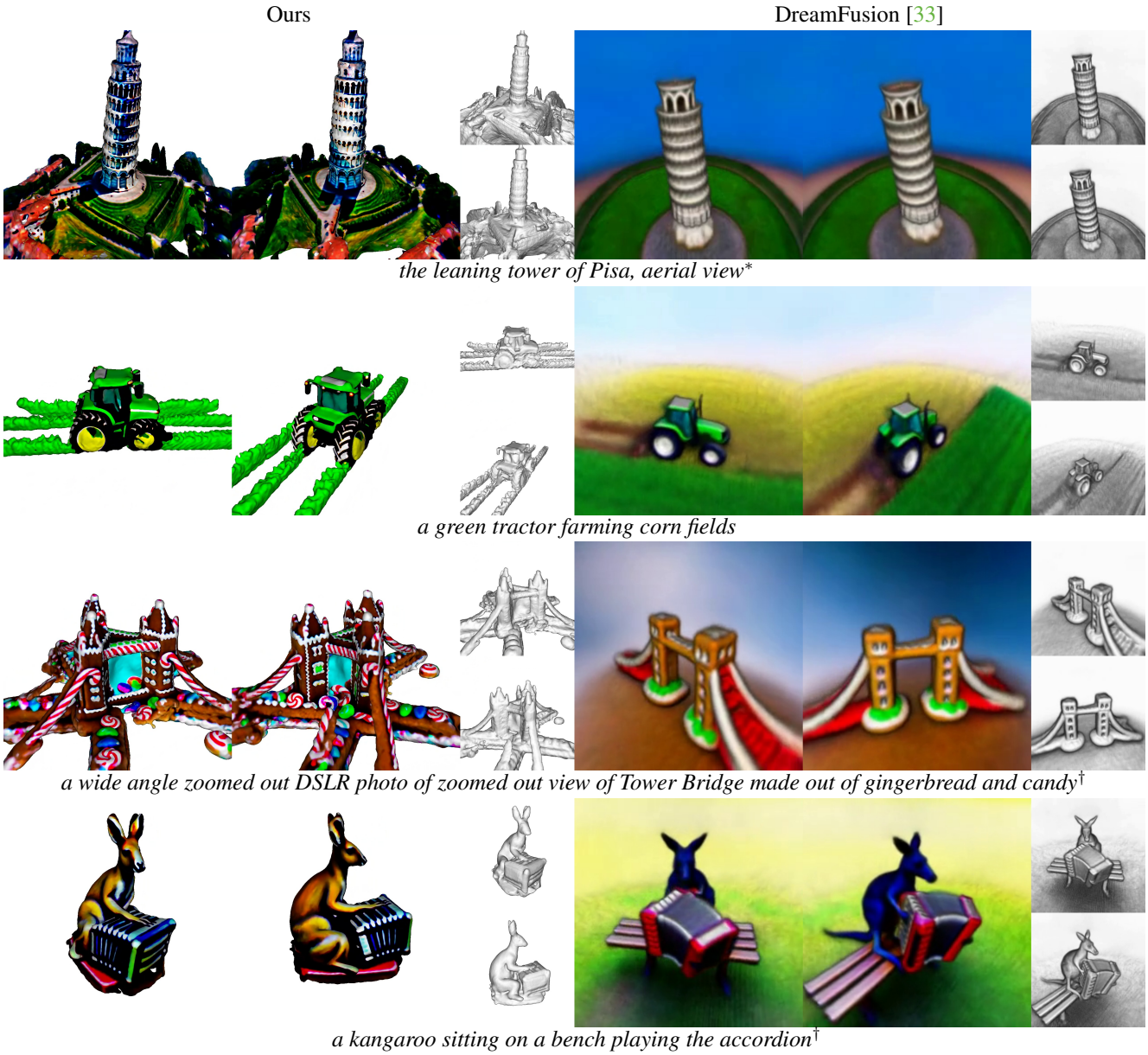


Figure 17. **Qualitative comparison with DreamFusion [33].** We use the same text prompt as in DreamFusion. For each 3D model, we render it from two views with a textureless rendering for each view and remove the background to focus on the 3D shape. For the DreamFusion results, we take frames from the videos published on the official webpage. `Magic3D` generates much higher quality 3D shapes on both geometry and texture compared with DreamFusion. $*$ *a DSLR photo of...* † *a zoomed out DSLR photo of...*

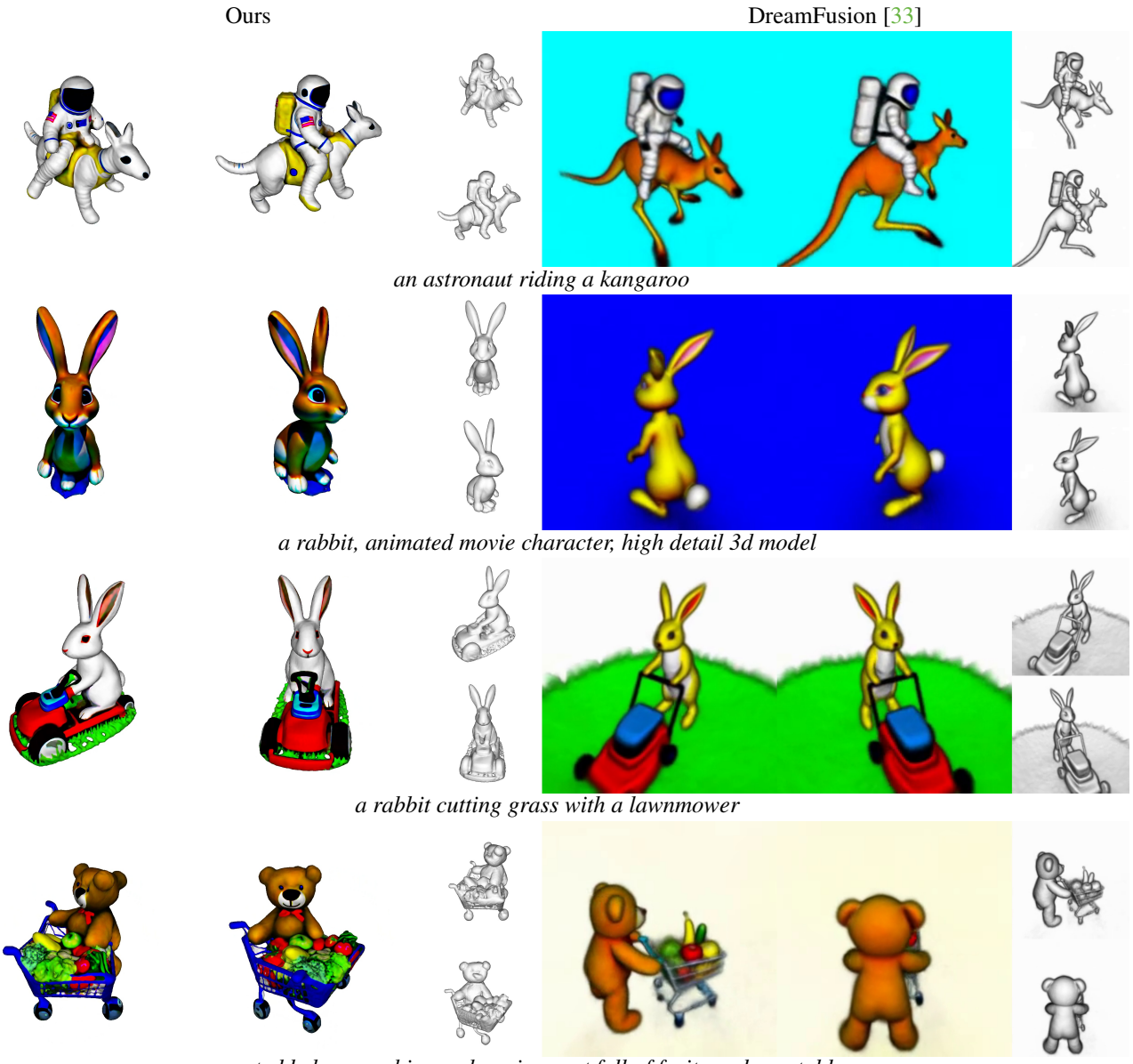


Figure 18. **Qualitative comparison with DreamFusion [33].** We use the same text prompt as in DreamFusion. For each 3D model, we render it from two views with a textureless rendering for each view and remove the background to focus on the 3D shape. For the DreamFusion results, we take frames from the videos published on the official webpage. `Magic3D` generates much higher quality 3D shapes on both geometry and texture compared with DreamFusion. $*$ *a DSLR photo of...* † *a zoomed out DSLR photo of...*