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
https://research.nvidia.com/labs/dir/magic3d

Abstract

DreamFusion [31] has recently demonstrated the utility of a pre-trained text-to-image diffusion model to optimize Neural Radiance Fields (NeRF) [23], 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× 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.

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.

Image content creation from text prompts [2, 28, 33, 36] has seen significant progress with the advances of diffusion models [13, 41, 42] 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, 47] 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. [47]. 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 [31] demonstrated its remarkable ability for text-conditioned 3D content generation by utilizing a pre-trained text-to-image diffusion model [36] 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) [23], 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 representation, 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 [25]. 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 $512 \times 512$. 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, 26] and make use of camera close-ups to recover high-frequency details in geometry and texture. As a result, our approach

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.
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, 35]. 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, 33, 34, 36]. Sampling images from diffusion models is time consuming. To generate high-resolution images, these models either utilize a cascade of super-resolution models [2, 36] or sample from a lower-resolution latent space and decode latents into high-resolution images [34]. 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, 20, 40, 45], pointclouds [1, 21, 46, 47, 49], meshes [9, 48], implicit [6, 22], 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 [23], recent works started investing in 3D-aware image synthesis [4, 5, 10, 11, 27, 29, 30, 38], 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, 29] 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 [37] 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 [32] 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 [31] showcased impressive capability in text-to-3D synthesis by utilizing a powerful pre-trained text-to-image diffusion model [36] 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 [31] 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 \( \theta \) is a coordinate-based MLP representing a 3D volume. The diffusion model \( \phi \) comes with a learned denoising function \( \epsilon_\phi(x_i; y, t) \) that predicts the sampled noise \( \epsilon \) given the noisy image \( x_i \), noise level \( t \), and text embedding \( y \). It provides the gradient direction to update \( \theta \) 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:

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

Here, \( w(t) \) is a weighting function. We view the scene model \( g \) and the diffusion model \( \phi \) as modular components of the
framework, amenable to choice. In practice, the denoising function $\epsilon_\phi$ 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 [36] 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 $64 \times 64$ 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 [31] 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 [36] 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 $64 \times 64$. In the second stage, we use the latent diffusion model (LDM) [34] that allows backpropagating gradients into rendered images at a high resolution $512 \times 512$; in practice, we choose to use the publicly available Stable Diffusion model [34]. Despite generating high-resolution images, the computation of LDM is manageable because the diffusion prior acts on the latent $z_t$ with resolution $64 \times 64$:

$$\nabla_\theta \mathcal{L}_{SDS}(\phi, g(\theta)) = \mathbb{E}_{t, \epsilon} \left[ w(t)(\epsilon_\phi(z_t; y, t) - \epsilon) \frac{\partial z}{\partial x} \frac{\partial x}{\partial \theta} \right].$$  (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 [31], 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 [25], 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 normals. We additionally maintain a spatial data structure that encodes scene occupancy and utilizes empty space skipping [19, 43]. Specifically, we use the density-based voxel pruning approach from Instant NGP [25] with an octree-based ray sampling and rendering algorithm [44]. 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., $512 \times 512$) 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 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, 39]. Each vertex $v_i \in V_T \subset \mathbb{R}^3$ contains a signed distance field (SDF) value $s_i \in \mathbb{R}$ and a deformation $\Delta v_i \in \mathbb{R}^3$ of the vertex from its initial canonical coordinate. Then, we extract a surface mesh from the SDF using a differentiable marching tetrahedra algorithm [39]. 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 [25], 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 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 10x 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 [18, 26]. We optimize both $s_i$ and $\Delta v_i$ for each vertex $v_i$, 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 [18]. 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 $\nabla_\theta \mathcal{L}_{\text{SDS}}$.

### 5. Experiments

We focus on comparing our method with DreamFusion [31] on 397 text prompts taken from the website of DreamFusion¹. 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

1https://dreamfusion3d.github.io/gallery.html
Figure 3. **Qualitative comparison with DreamFusion** [31]. 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.

Table 1. **User preference studies.** We conducted user studies to measure preference for 3D models generated using 397 prompts released by DreamFusion. Overall, 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.

<table>
<thead>
<tr>
<th>Comparison</th>
<th>Preference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magic3D vs. DreamFusion [31]</td>
<td></td>
</tr>
<tr>
<td>• More realistic</td>
<td>58.3%</td>
</tr>
<tr>
<td>• More detailed</td>
<td>66.0%</td>
</tr>
<tr>
<td>• More realistic &amp; detailed</td>
<td>61.7%</td>
</tr>
<tr>
<td>Magic3D vs. Magic3D (coarse only)</td>
<td>87.7%</td>
</tr>
</tbody>
</table>

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.

**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
Coarse NeRF
Fine-tuned NeRF
Fine-tuned Mesh

baby dragon hatching out of a stone egg∗ bagel filled with cream cheese and lox∗
a marble bust of a mouse a ladybug†

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.

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.

**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 memory-efficient sparse 3D representation as the ideal candidate for the scene model. However, rendering $512 \times 512$ 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 $512 \times 512$ as input to the LDM. We find it generates objects with worse shapes. Fig. 4 shows two examples with scene rendering resolution $64 \times 64$ and $256 \times 256$ 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 $256 \times 256$ 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 [35] 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.

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 \([34]\), 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 \(1 \times 10^{-5}\) for 1,500 iterations; for LDM, we fine-tune with learning rate \(1 \times 10^{-6}\) 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.

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. 7, 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, Seunjun 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


[27] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised...


