

# HiFA: High-fidelity Text-to-3D with Advanced Diffusion Guidance

Junzhe Zhu

Stanford University

josefzhu@stanford.edu

Peiyue Zhuang

Stanford University

peiyue@stanford.edu

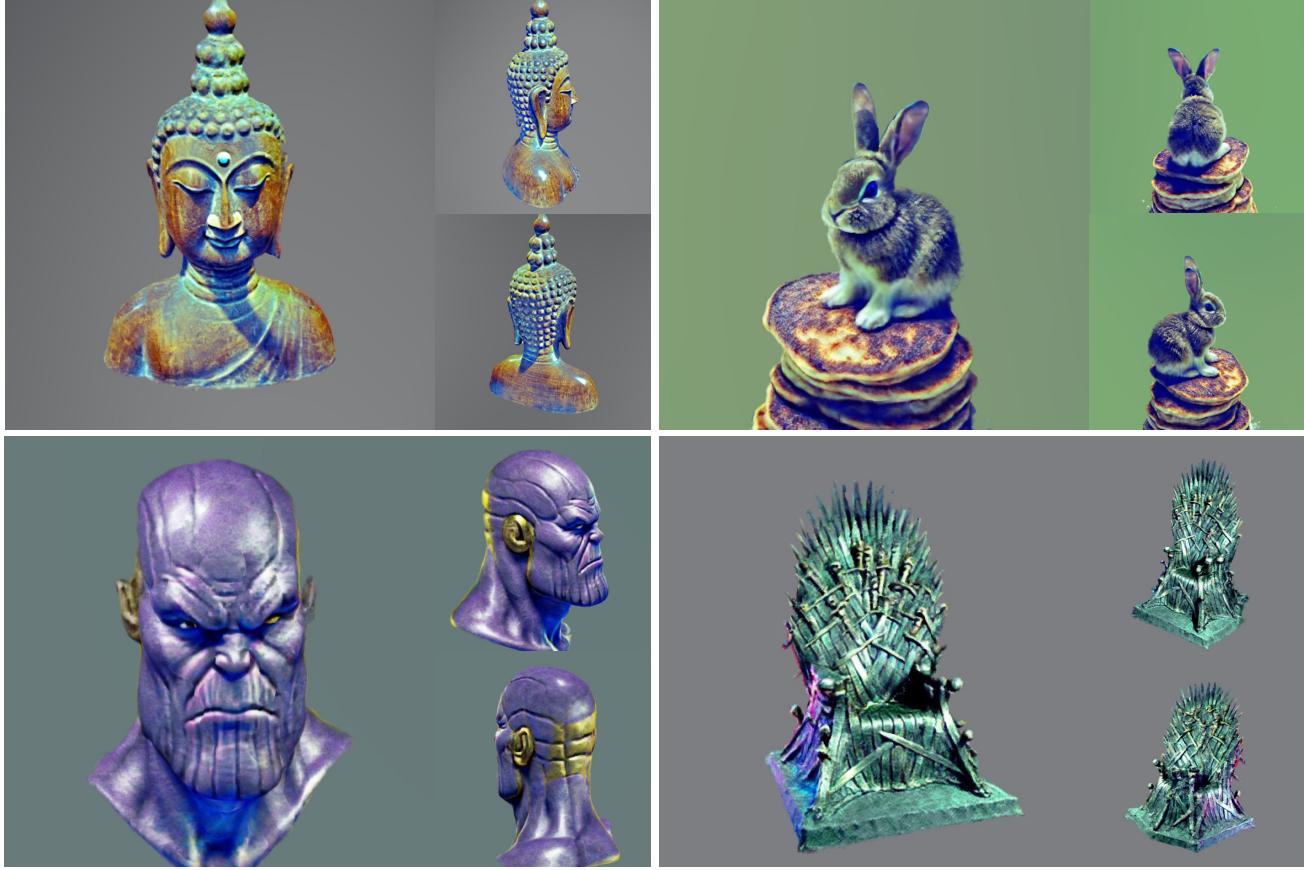


Figure 1. Visualization of our multi-view synthesis results given text prompts. The prompts, arranged from left to right and top to bottom, include (1) A wooden buddha with just one face; (2) A baby bunny sitting on top of a stack of pancakes; (3) the head of Thanos; (4) the iron throne from game of thrones.

## Abstract

Automatic text-to-3D synthesis has achieved remarkable advancements through the optimization of 3D models. Existing methods commonly rely on pre-trained text-to-image generative models, such as diffusion models, providing scores for 2D renderings of Neural Radiance Fields (NeRFs) and being utilized for optimizing NeRFs. However, these methods often encounter artifacts and inconsis-

tencies across multiple views due to their limited understanding of 3D geometry. To address these limitations, we propose a reformulation of the optimization loss using the diffusion prior. Furthermore, we introduce a novel training approach that unlocks the potential of the diffusion prior. To improve 3D geometry representation, we apply auxiliary depth supervision for NeRF-rendered images and regularize the density field of NeRFs. Extensive experiments demonstrate the superiority of our method over prior works, re-



Figure 2. We show our results of Text-to-3D synthesis rendered from 3 views. The text prompts, arranged from left to right and top to bottom, include: (1) a beautifully carved wooden queen chess piece; (2) A beautiful dress made out of garbage bags, on a mannequin. Studio lighting, high quality, high resolution; (3) a silver candelabra sitting on a red velvet tablecloth, only one candle is lit; (4) small saguaro cactus planted in a clay pot; (5) a high pile of chocolate chip cookies; (6) a kingfisher bird; (7) sorting hat from harry potter; (8) DSLR photo of a beautiful peacock with long neck.

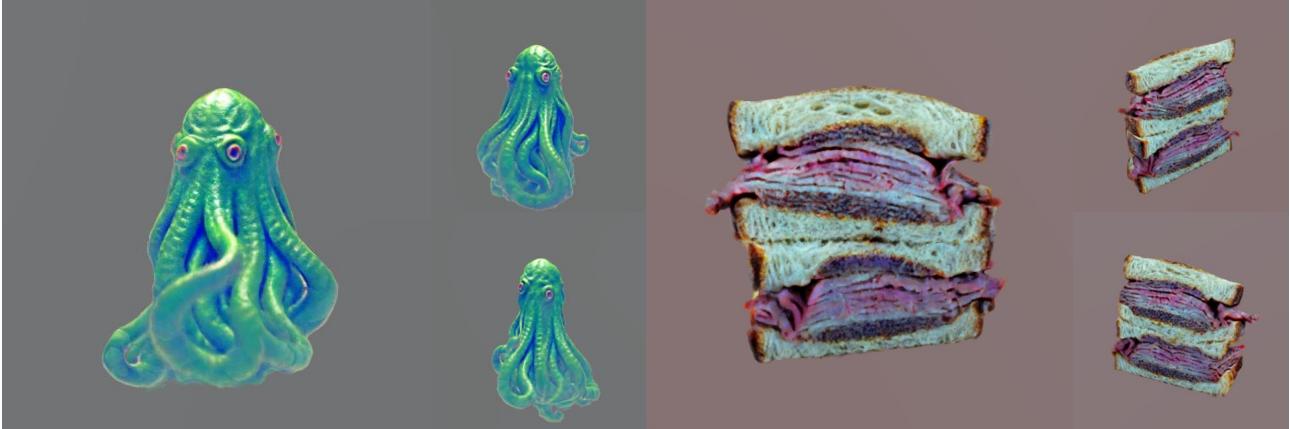


Figure 3. We show our results of Text-to-3D synthesis rendered from 3 views. The text prompts, arranged from left to right, include: (1) a DSLR photo of cthulhu; (2) an overstuffed pastrami sandwich.



Figure 4. We show our rendered images (*column 2 and 4*) of 3D assets with their corresponding meshes (extracted using marching cubes from NeRF density fields) on their left (*column 1 and 3*). The text prompts, arranged from left to right and top to bottom, include: (1) A baby bunny sitting on top of a stack of pancakes; (2) DSLR photo of a beautiful peacock with long neck; (3) a beautifully carved wooden queen chess piece; (4) game asset of a leather shoe made from dragon skin; (5) a kingfisher bird (6) head of thanos; (7) a high pile of chocolate chip cookies; (8) small saguaro cactus planted in a clay pot.

sulting in advanced photo-realism and improved multi-view consistency.

## 1. Introduction

An automatic text-to-3D synthesis task aims to generate 3D assets from a text description. It has gained significant attention due to its applications in digital content generation, film-making, and Virtual Reality (VR) [1, 8]. However, text-to-3D synthesis is challenging due to the scarcity of large-scale annotated 3D datasets and the need for efficient 3D generative models. Recent advancements have emerged to address these challenges by leveraging pre-trained 2D models as priors for optimizing a 3D representation model [1, 5, 6, 8, 16, 28, 29, 32]. Commonly, pre-trained CLIP [17] or 2D diffusion generative models on images [22, 23] are applied to optimize Neural radiance fields (NeRFs) [11] or meshes [8].

Concretely, in prior work [16], a diffusion model produces a denoising score, specifically the noise residual, for a noisy NeRF-rendered image. Subsequently, this denoising score is utilized in back-propagation to optimize NeRFs. Since the denoising score is obtained through the diffusion model, the backward gradients inevitably incorporate a Jacobian term associated with the diffusion model. To circumvent the back-propagation through the diffusion model, existing approaches either directly exclude the Jacobian term associated with the diffusion model [8, 16] or utilize Monte-Carlo estimation to compute the expectation of scores from multiple noisy NeRF-rendered images [29]. While these methods manage to produce plausible results, we have observed that they are susceptible to artifacts and exhibit issues of multi-view inconsistency in rendered images [8, 16, 28, 29]. We suspect that these methods have not fully unlocked the potential of 2D diffusion prior. In particular, we observed that prior work [16, 29] randomly sampling a noise level in the diffusion process for optimizing NeRFs is suboptimal. It leads to out-of-distribution (OOD) issues with input of the diffusion model during initial training iterations and divergence issues with output of the diffusion model during end training iterations.

Our work focuses on enhancing *photo-realism* and *multi-view consistency* of text-to-3D synthesis. We adopt a similar setup as prior work [8, 16, 28, 29], employing a pre-trained 2D diffusion model into a 3D generative model of radiance fields. To address the aforementioned issues, we introduce a new optimization approach that utilizes the reformulated diffusion guidance. Concretely, instead of using the noise residual of the diffusion model as the score, we re-derive the score as the image residual (or latent vector residual when using a latent diffusion model [22]). Moreover, rather than randomly sampling a time step (referred to as a noise level) of the diffusion process, we propose a time annealing ap-

proach that gradually decreases the time steps during training iterations. In addition, we propose auxiliary supervision on rendered images (and/or latent vectors). To improve the representation of 3D geometry, we incorporate depth supervision and apply regularization techniques to refine the volume density fields of a NeRF. We summarize our technical contributions as follows:

- We propose a new optimizing approach by reformulating the denoising score for rendered images, accompanied by auxiliary supervision on rendered images (and/or latent vectors).
- We introduce a new time step annealing approach that progressively decreases the time steps during training iterations.
- We incorporate depth supervision and regularization of volume density fields of a NeRF.

Please visit our [website](#) for additional results.

## 2. Related Work

**NeRFs** [11] are a family of implicit neural representation methods designed to encode the shape and/or appearance of 3D objects using a parameterized neural network. NeRFs enable impressive novel-view synthesis via volume rendering. In contrast to voxel-based [13, 21, 31] and mesh-based methods [15, 30], NeRFs represent 3D objects in continuous space and free from the constraints of a specific object topology. In our work, we employ NeRFs for 3D asset representation.

**Diffusion models** [3], belonging to the broad family of denoising score-matching generative models [25], have achieved remarkable success in a variety of applications such as image editing [10], text-to-image synthesis [18, 19, 22, 23], text-to-video synthesis [2], text-to-3D synthesis [1, 5, 6, 8, 16, 28], and continuous field synthesis [33]. Moreover, to address the challenge of slow sampling speed resulting from the iterative generation process of diffusion models, researchers have proposed efficient sampling, classified into learning-free [9, 26] and learning-based [24, 27] methods. In our work, we utilize a pre-trained StableDiffusion model [22] to incorporate image priors for NeRF optimization.

**Text-to-3D synthesis** approaches commonly employ image priors to optimize 3D models. These approaches can be roughly categorized into two groups: (i) CLIP-guided text-to-3D approaches [5, 6] that utilize pre-trained cross-modal matching models, CLIP [17], and (ii) diffusion-guided text-to-3D approaches [1, 8, 16, 28] that utilize 2D diffusion models such as Imagen [23] and StableDiffusion [22]. We follow the diffusion-guided methods due to their superior performance in text-to-3D synthesis.

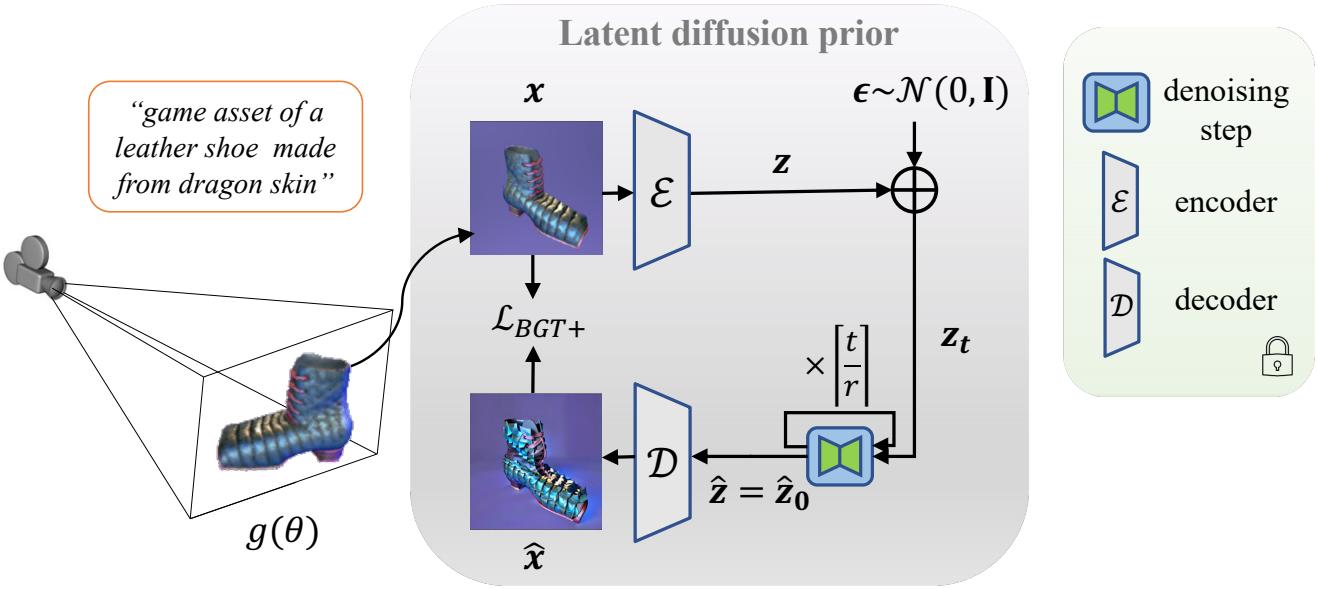


Figure 5. Overview of our proposed method for text-to-3D synthesis. We aim to optimize a 3D model  $g(\theta)$  using a pre-trained 2D latent diffusion prior. To achieve this, we employ a latent diffusion model to provide gradients. Specifically, the diffusion model takes a rendered image  $\mathbf{x}$  as input and provides the estimate of the input rendered image, denoted as  $\hat{\mathbf{x}}$ . Unlike existing works [16] that solely focus on computing noise residuals in the low-resolution latent space of the diffusion model, we propose a novel loss  $\mathcal{L}_{BGT+}$  that computes a higher-resolution image residual.

Specifically, Dreamfusion [16] first introduced a Score Distillation Sampling (SDS) loss derived from the distillation of Imagen [23]. SDS minimizes the Kullback-Leibler (KL) divergence between a Gaussian noise distribution and the estimated noise distribution. SDS is widely applied in later text-to-3D synthesis work [1, 8, 28]. Score-Jacobian-Chaining [29] proposed Perturb-and-Average Scoring method to aggregate 2D image gradients of StableDiffusion [22] over multiple viewpoints into a 3D asset gradient. However, due to a lack of 3D structure understanding when lifting 2D diffusion models to 3D, these text-to-3D synthesis methods often result in difficulties in achieving realistic 3D assets in terms of multi-view consistency and photo-realism. In this work, we aim to address these issues by introducing a new optimization approach and incorporating additional supervision on rendered images, depth, and volume densities of NeRFs.

### 3. Preliminary: Score Distillation Sampling

Recent works on text-to-3D synthesis [8, 16, 29] have focused on optimizing a 3D model using 2D diffusion prior. In these works, a 3D model is parameterized by  $\theta$ , denoted as  $\mathbf{x} = g(\theta)$ , which allows for the synthesis of an image  $\mathbf{x}$  based on a given camera pose. The 2D diffusion model contains a denoising autoencoder  $\epsilon_\phi$  which predicts the sampled noise  $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ , denoted as  $\hat{\epsilon}_t$ , by employing the noisy image  $\mathbf{x}_t$ , a time step  $t$ , a text embedding  $\mathbf{y}$ . For-

mally, we have  $\hat{\epsilon}_t := \epsilon_\phi(\mathbf{x}_t; \mathbf{y}; t)$ . To simplify the notion, we will omit the sub-index  $t$  and instead use  $\hat{\epsilon}$  to represent  $\hat{\epsilon}_t$ . The diffusion model provides gradients for updating the 3D model  $g(\theta)$ :

$$\begin{aligned} & \nabla_\theta \mathcal{L}_{\text{diff}}(\phi, \mathbf{x}) \\ &= \nabla_\theta \mathbb{E}_{t, \epsilon} [\omega(t) \|\hat{\epsilon} - \epsilon\|^2] \\ &= \mathbb{E}_{t, \epsilon} [\omega(t) (\hat{\epsilon} - \epsilon) \frac{\partial \hat{\epsilon}}{\partial \mathbf{x}_t} \frac{\partial \mathbf{x}_t}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial \theta}], \end{aligned} \quad (1)$$

where  $\omega(t)$  is a weighting function.

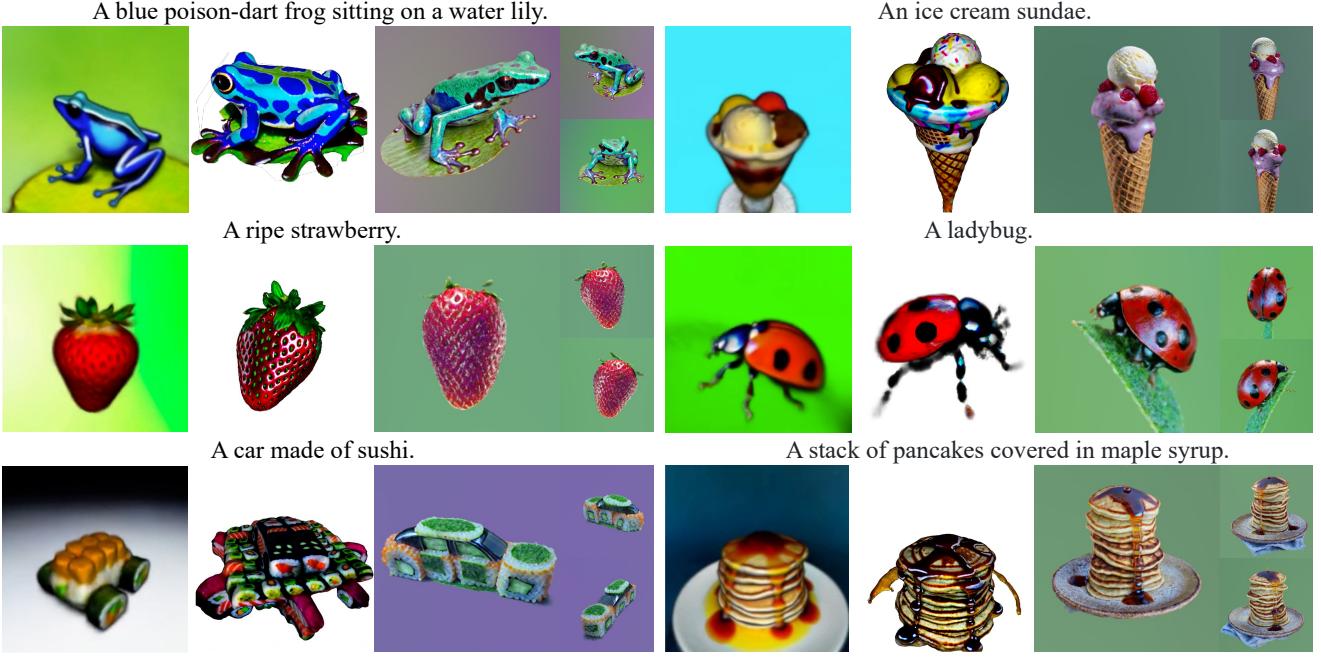
Dreamfusion [16] omitted the U-Net Jacobian term  $\frac{\partial \hat{\epsilon}}{\partial \mathbf{x}_t}$  and the scalar term  $\frac{\partial \mathbf{x}_t}{\partial \mathbf{x}}$  in Eq. 1. The new loss is called score distillation sampling (SDS). Consequently, the gradient of the SDS loss can be written as:

$$\nabla_\theta \mathcal{L}_{\text{SDS}}(\phi, \mathbf{x}) = \mathbb{E}_{t, \epsilon} [\omega(t) (\hat{\epsilon} - \epsilon) \frac{\partial \mathbf{x}}{\partial \theta}]. \quad (2)$$

### 4. Method

We aim to produce photo-realistic and multi-view consistent 3D models driven by text prompts. For this, we propose our method, as illustrated in Fig. 5, for which we provide an overview next.

In our work, we employ a *latent* diffusion model as a diffusion prior, specifically utilizing a publicly available StableDiffusion model (SD) [22]. In Sec. 4.1, we present



Dreamfusion

Magic3D

Ours

Dreamfusion

Magic3D

Ours

Figure 6. Qualitative text-to-3D synthesis comparison with Dreamfusion [16] and Magic3D [8]. The text prompt is displayed on top of the rendered images. Our proposed method provides significantly improved 3D texture quality and more realistic lighting.



Figure 7. Qualitative comparison with Score-Jacobian-Chaining (SJC) [29]. Our method (*bottom*) achieves more appealing renderings.

our formulation of the SDS loss where the U-Net Jacobian term *does not* appear. Additionally, we introduce regularization techniques in both the latent and image space of SD, which seamlessly complement the use of our formulated SDS. Moreover, we analyze the training challenges encountered in previous works [16, 29] such as an out-of-distribution issue of diffusion input and a divergence issue of diffusion output. To address these issues, we propose a

new training approach that incorporates annealed time step scheduling. Furthermore, in Sec. 4.2, we enhance the NeRF representation by introducing depth prior supervision and auxiliary regularization, specifically targeting the variance of sampled coordinates along rays (we name it *z-variance*) in NeRFs.

#### 4.1. Advancing training process with reformulated SDS

We employ a pre-trained SD model [22] as 2D diffusion prior. Since the SD model performs the diffusion process in the *latent* space, it results in changes to the SDS gradients in this scenario. Therefore, we first derive the original SDS gradient within the latent space of SD, followed by its reformulation. Specifically, an SD model consists of an encoder  $\mathcal{E}$ , a decoder  $\mathcal{D}$ , and a denoising autoencoder  $\epsilon_\phi$ . The encoder  $\mathcal{E}$  compresses the input image  $x$  into a latent vector  $z$  with resolution  $64 \times 64$ , written as  $z = \mathcal{E}(x)$  and the decoder  $\mathcal{D}$  reconstructs a latent vector back to an image  $x = \mathcal{D}(z)$ .

An input of the autoencoder  $\epsilon_\phi$  is a noisy latent vector  $z_t$  at noise level  $t$ . The noisy latent vector  $z_t$  is denoted as

$$z_t = \sqrt{\bar{\alpha}_t} z + \sqrt{1 - \bar{\alpha}_t} \epsilon, \quad (3)$$

where  $\epsilon \sim \mathcal{N}(0, I)$ . The weight is defined as  $\bar{\alpha}_t := \prod_{s=1}^t \alpha_s$ , where  $\alpha_s$  is a variance schedule for adding Gaus-

sian noise to the data during the diffusion process (refer to [3] for details). The denoising autoencoder  $\epsilon_\phi$  predicts the original noise at noise level  $t$ , denoted as  $\hat{\epsilon}$  (we omit the sub-index  $t$  as before):

$$\hat{\epsilon} := \epsilon_\phi(z_t; y; t). \quad (4)$$

Subsequently, the SDS gradient in the latent space of SD [22] is written as

$$\nabla_\theta \mathcal{L}_{\text{SDS}}(\phi, z) = \mathbb{E}_{t, \epsilon} [\omega(t)(\hat{\epsilon} - \epsilon) \frac{\partial z}{\partial \theta}]. \quad (5)$$

To our knowledge, the concrete formulation of this loss  $\mathcal{L}_{\text{SDS}}$  remains unknown.<sup>1</sup>

In contrast, we can formulate a concrete definition for  $\mathcal{L}_{\text{SDS}}$  by changing noise residuals,  $(\hat{\epsilon} - \epsilon)$ , to latent vector residuals. Specifically, we have

$$\mathcal{L}_{\text{SDS}}(\phi, z) = \mathbb{E}_{t, \epsilon} [\omega(t) \frac{\sqrt{\bar{\alpha}_t}}{2\sqrt{1-\bar{\alpha}_t}} \|z - \hat{z}_{\text{1step}}\|^2], \text{ where} \quad (6)$$

$$\hat{z}_{\text{1step}} = \frac{1}{\sqrt{\bar{\alpha}_t}} (z_t - \sqrt{1-\bar{\alpha}_t} \epsilon_\phi(z_t; y, t)). \quad (7)$$

$\hat{z}_{\text{1step}}$  is the denoised  $z$  estimate, obtained by calling upon the denoising autoencoder  $\epsilon_\phi$  once and  $(\hat{z}_{\text{1step}} - z)$  is the latent vector residual. Note that  $\hat{z}_{\text{1step}}$  are from the distribution generated by  $z, t$  and  $\epsilon$ . A concrete formulation of the loss  $\mathcal{L}_{\text{SDS}}$ , as presented in Eq. 6, can facilitate loss analysis and gradient computation in practical scenarios. Consequently, the gradient of  $\mathcal{L}_{\text{SDS}}(\phi, z)$  in Eq. 6 can be written as

$$\nabla_\theta \mathcal{L}_{\text{SDS}}(\phi, z) = \mathbb{E}_{t, \epsilon} [\omega(t) \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}} (z - \hat{z}_{\text{1step}}) \frac{\partial z}{\partial \theta}] \quad (8)$$

*Proof:* we prove that the two SDS gradients in Eq. 5 and Eq. 8 are equivalent.

$$\begin{aligned} & \nabla_\theta \mathcal{L}_{\text{SDS}}(\phi, z) \\ &= \omega(t) \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}} (z - \hat{z}_{\text{1step}}) \frac{\partial z}{\partial \theta} \quad (\text{Eq. 8}) \\ &= \omega(t) \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}} \left( z - \frac{1}{\sqrt{\bar{\alpha}_t}} (z_t - \sqrt{1-\bar{\alpha}_t} \epsilon_\phi(z_t; y, t)) \right) \frac{\partial z}{\partial \theta} \\ &= \omega(t) \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}} \left( \frac{\sqrt{1-\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_t}} (\hat{\epsilon} - \epsilon) \right) \frac{\partial z}{\partial \theta} \\ &= \omega(t) (\hat{\epsilon} - \epsilon) \frac{\partial z}{\partial \theta} \quad (\text{Eq. 5}) \end{aligned}$$

Therefore, we introduce a specific formulation for the loss  $\mathcal{L}_{\text{SDS}}$ , which yields identical gradients as in Eq. 5, while

<sup>1</sup>This brings challenges for loss visualization and analysis, and back-propagation when using deep learning frameworks like PyTorch.

also benefiting from the advantages associated with having a concrete formulation. Moreover, to intuitively understand Eq. (6-8), we consider the  $\hat{z}_{\text{1step}}$  as a constant that is independent of  $z$ ; in practice, detach it from the computation graph. In this context,  $\hat{z}_{\text{1step}}$  is a *bootstrapped ground truth* as it is the one-step estimate of  $z$  obtained using the denoising autoencoder  $\epsilon_\phi$ .

Next, we analyze training challenges of prior work [16, 29] and propose two approaches to address those.

**Training challenges of prior work** include an out-of-distribution (OOD) issue of the diffusion input and a divergence issue of the diffusion output, due to random time step sampling in prior works [16, 29]. Firstly, during the initial training iterations, rendered images (and latent vectors) contain artifacts, which lie outside the original distribution of diffusion input. Particularly, when the randomly sampled time step  $t$  is small, the step size of denoising is relatively small, as well. As a result, the recovered latent vector  $\hat{z}$ , close to the OOD input, also falls outside the original distribution of latent vectors. Secondly, another significant issue is the divergence of predicted images produced by the diffusion model. Towards the end of the training iterations, a well-trained NeRF renders images of an *almost determined* 3D asset. However, when a large sampled time step  $t$  is used, the denoised image predicted by the diffusion model can appear *distinct* and *unrelated* to the given input rendering. It leads to the generation of inaccurate gradients from the diffusion model. Consequently, these inaccurate gradients are detrimental to the optimization of the 3D model.

To address these issues, we introduce two approaches: (i) instead of using  $\hat{z}_{\text{1step}}$  for SDS, we employ a more accurate estimation for the latent vector  $z$ , denoted as  $\hat{z}$ , obtained through iterative denoising, and (ii) we gradually decrease the intensity of the added noise during training by annealing the step  $t$ . In the following, we provide details for each.

**Iterative estimation of  $\hat{z}$ .** We introduce an enhanced formulation of the SDS loss (Eq. 6) by replacing  $\hat{z}_{\text{1step}}$  with  $\hat{z}$ , where the latent vector  $\hat{z}$  is better estimation of the original  $z$  via iterative denoising. This adapted loss of Eq. 6 is denoted as  $\mathcal{L}_{\text{BGT}}$ , formally written as

$$\mathcal{L}_{\text{BGT}}(\phi, z, \hat{z}) = \mathbb{E}_{t, \epsilon} \|z - \hat{z}\|^2. \quad (9)$$

Iterative estimation of  $\hat{z}$  refers to gradually denoise the noisy latent vector  $z_t$  until  $t = 0$ . Specifically, we initiate the iterative estimation process with a step ratio  $r$  by obtaining  $\hat{z}_{t-r}, \hat{z}_{t-2r}, \dots$ , iteratively until reaching  $\hat{z}_0$ , which we use as the ground truth, i.e.,  $\hat{z} := \hat{z}_0$ . To accelerate this reverse diffusion process, many sampling approaches, such as DDIM [26], can be adapted here. We take DDIM [26] as an example, we iteratively obtain a denoised latent vector

---

**Algorithm 1** Training Procedure

---

**Input:** A pre-trained SD [22] consisting of an encoder  $\mathcal{E}$ , a decoder  $\mathcal{D}$ , and a denoising autoencoder  $\epsilon_\phi$ ; a rendering  $\mathbf{x} = g(\theta)$ ; a latent vector  $\mathbf{z} = \mathcal{E}(\mathbf{x})$ ; a number of total training steps `total_iter`; range of the diffusion time steps  $[t_{\max}, t_{\min}]$ ; a conditioning  $\mathbf{y}$ ; scaling coefficients  $\{\bar{\alpha}_t\}$ ; DDIM's hyper-parameter  $\eta$  and step ratio  $r$ .

- 1: **for** `iter` = [0, `total_iter`] **do**
- 2:    $t = t_{\max} - (t_{\max} - t_{\min}) \sqrt{\frac{\text{iter}}{\text{total\_iter}}}$
- 3:    $\mathbf{z}_t = \hat{\mathbf{z}}_t = \sqrt{\bar{\alpha}_t} \mathbf{z} + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}$ , where  $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$
- 4:    $i = t$
- 5:   **do**
- 6:      $i_{\text{prev}} = \max(0, i - r)$
- 7:      $\sigma_i = \eta \sqrt{\frac{1 - \alpha_{i_{\text{prev}}}}{1 - \alpha_i}} \sqrt{\frac{i - \alpha_i}{\alpha_{i_{\text{prev}}}}}$
- 8:      $\hat{\mathbf{z}}_{\text{prev}} = \sqrt{\bar{\alpha}_{i_{\text{prev}}}} \left( \frac{\hat{\mathbf{z}}_i - \sqrt{1 - \bar{\alpha}_i} \epsilon_\phi(\hat{\mathbf{z}}_i; \mathbf{y}, i)}{\sqrt{\bar{\alpha}_i}} \right)$
- 9:      $+ \sqrt{1 - \bar{\alpha}_{i_{\text{prev}}} - \sigma_i^2} \cdot \epsilon_\phi(\hat{\mathbf{z}}_i; \mathbf{y}, i) + \sigma_i \boldsymbol{\epsilon}$
- 10:     $i = i_{\text{prev}}$
- 11:     $\hat{\mathbf{z}}_i = \hat{\mathbf{z}}_{\text{prev}}$
- 12:   **while**  $i > 0$
- 13:    $\hat{\mathbf{z}} = \hat{\mathbf{z}}_0$  and  $\hat{\mathbf{x}} = \mathcal{D}(\hat{\mathbf{z}})$
- 14:   Compute the loss gradient  $\nabla_\theta \mathcal{L}$  and update  $\theta$
- 15: **end for**

**Return:**  $\theta$

---

$\hat{\mathbf{z}}_{i-r}$  from  $\hat{\mathbf{z}}_i$ :

$$\begin{aligned} \hat{\mathbf{z}}_{i-r} &= \sqrt{\bar{\alpha}_{i-r}} \left( \frac{\hat{\mathbf{z}}_i - \sqrt{1 - \bar{\alpha}_i} \epsilon_\phi(\hat{\mathbf{z}}_i; \mathbf{y}, i)}{\sqrt{\bar{\alpha}_i}} \right) \\ &+ \sqrt{1 - \bar{\alpha}_{i-r} - \sigma_i^2} \cdot \epsilon_\phi(\hat{\mathbf{z}}_i; \mathbf{y}, i) + \sigma_i \boldsymbol{\epsilon}, \end{aligned} \quad (10)$$

where  $\alpha_i$  is a variance schedule function,  $\sigma_i = \eta \sqrt{\frac{1 - \alpha_{i-r}}{1 - \alpha_i}} \sqrt{\frac{1 - \alpha_i}{\alpha_{i-r}}}$ , and  $\eta$  is a stochasticity hyperparameter.

After obtaining the estimated latent vector  $\hat{\mathbf{z}}$ , a further adapted loss can be naturally obtained, by incorporating additional supervision for recovered images. Formally, we define the adapted loss  $\mathcal{L}_{\text{BGT+}}$  as

$$\mathcal{L}_{\text{BGT+}}(\phi, \mathbf{z}, \hat{\mathbf{z}}) = \mathbb{E}_{t, \epsilon} [\|\mathbf{z} - \hat{\mathbf{z}}\|^2 + \lambda_{\text{rgb}} \|\mathbf{x} - \hat{\mathbf{x}}\|^2], \quad (11)$$

where  $\hat{\mathbf{x}}$  is an recovered image obtained through the decoder  $\mathcal{D}$ , formally written as  $\hat{\mathbf{x}} = \mathcal{D}(\hat{\mathbf{z}})$ .  $\lambda_{\text{rgb}}$  is a scaling parameter. We mention that when utilizing a pre-trained latent diffusion model, the latent vectors are typically at a low resolution. In contrast, the images  $\mathbf{x}$  and  $\hat{\mathbf{x}}$  are at a higher resolution ( $64 \times 64$  and  $512 \times 512$ , respectively in the case of using StableDiffusion [22]). We ablate these proposed loss terms in Sec. 5. Note that computing the higher-resolution RGB loss is not possible under prior work's formulation of

the gradient  $\nabla_\theta \mathcal{L}_{\text{SDS}}$  (Eq. 8), since the denoising autoencoder  $\epsilon_\phi$  is restricted to latent space only.

**Annealed time step scheduling** is an alternative to replace random time step sampling in the training process, resulting in enhanced performance. Specifically, we introduce a high time step  $t$  to the rendered image during the *initial* training iterations. This intentional noise allows the image to align with the distribution modeled by the diffusion prior. As the training progresses, we gradually reduce the time step  $t$ , enabling the rendered image to capture finer details by leveraging more stable gradients with lower variance.

In practice, we pick the schedule

$$t = t_{\max} - (t_{\max} - t_{\min}) \sqrt{\frac{\text{iter}}{\text{total\_iter}}}, \quad (12)$$

where  $t$  decreases rapidly at the beginning of the training process and slows down toward the end. This scheduling strategy allocates more training steps to lower values of  $t$ , ensuring that fine-grained details can be adequately learned during the latter stages of training. Empirically, we choose  $t_{\max} = 0.98$  and  $t_{\min} = 0.2$ . We find annealing the timestep to be critical.

## 4.2. Advanced supervision for NeRFs

A NeRF renders a color  $\hat{C}_r$ , denoted as  $\hat{C}_r = \sum_{i=1}^N w_i c_i$ , where  $w_i$  and  $c_i$  are respectively estimated weights and colors of the sampled coordinate  $z_i$  along a ray  $r$  (details explained in [11]).  $N$  refers to the number of sampled points along the ray  $r$ . Subsequently, we compute the depth value  $\mu_{z_r}$  and the disparity value  $d_{z_r}$  of the ray  $r$  as

$$\mu_{z_r} = \sum_i z_i \frac{w_i}{\sum_i w_i}, \text{ and } d_{z_r} = \frac{1}{\mu_{z_r}}, \quad (13)$$

where  $\frac{w_i}{\sum_i w_i}$  can be considered as a probability mass function.

**Depth prior supervision** is adopted into NeRFs to ensure multi-view consistency. NeRFs are known to be able to render realistic images with implausible underlying geometry [14]. Since our 2D guidance does not have spatial consistency built into it, we propose to use pre-trained depth prediction models as additional guidance to alleviate this issue. To simplify the notation, we denote rendered disparity map as  $d$ .

We apply a regularization loss  $\mathcal{L}_d$  on the disparity map  $d$ , as shown in Fig. 8. Specifically, we employ a pre-trained depth predictor [20] to estimate the *pseudo ground truth* of the disparity map  $d^*$ , for the rendered image  $\mathbf{x}$  and compute the depth loss  $\mathcal{L}_d$ . Formally, we have

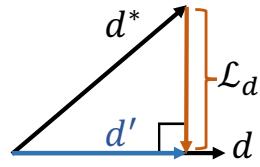


Figure 8. Illustration of the depth loss  $\mathcal{L}_d$ .

$$\mathbf{d}' = \frac{\langle \mathbf{d}, \mathbf{d}^* \rangle \mathbf{d}}{\|\mathbf{d}\|^2}, \text{ and } \mathcal{L}_d = \mathbb{E}_{\mathbf{d}} \|\mathbf{d}' - \mathbf{d}^*\|^2 \quad (14)$$

Note that we normalize the disparity maps  $\mathbf{d}$  and  $\mathbf{d}^*$ .

**Regularization for z-variance** aims to reduce the variance of the distribution of the sampled z-coordinates  $z_i$  along the ray  $r$ . A smaller variance indicates crisper surfaces in geometry. To this end, we compute the z-variance along the ray  $r$ , denoted as  $\sigma_{z_r}^2$ :

$$\sigma_{z_r}^2 = \mathbb{E}_{z_i}[(z_i - \mu_{z_r})^2] = \sum_i (z_i - \mu_{z_r})^2 \frac{w_i}{\sum_i w_i}. \quad (15)$$

The regularization loss  $\mathcal{L}_{zvar}$  for the variance  $\sigma_{z_r}^2$  is defined as

$$\begin{aligned} \mathcal{L}_{zvar} &= \mathbb{E}_r \delta_r \sigma_{z_r}^2, \\ \delta_r &= 1 \text{ if } \sum_i w_i > 0.5, \text{ else } 0. \end{aligned} \quad (16)$$

Here,  $\delta_r$  serves as an indicator function (or binary weight) for filtering out background rays. We find this loss to be immensely useful in ensuring the geometrical consistency of our 3D model. Removing the loss  $\mathcal{L}_{zvar}$ , as illustrated in Fig. 10, results in the cloudy and less accurate geometry.

We define our total loss function as

$$\mathcal{L} = \mathcal{L}_{BGT+} + \lambda_d \mathcal{L}_d + \lambda_{zvar} \mathcal{L}_{zvar}, \quad (17)$$

where  $\lambda_d$  and  $\lambda_{zvar}$  are loss weights. We present our training procedure in Algorithm 1.

## 5. Experiments

We present our implementation details in Sec. 5.1 and evaluate our method to generate 3D assets from text prompts in Sec. 5.2. Specifically, we compare our method with state-of-the-art text-to-3D synthesis methods including Dreamfusion [16], Magic3D [8], and Score-Jacobian-Chaining [29]. Moreover, we conduct extensive ablation studies to verify the effectiveness of each proposed technique. Finally, we analyze the limitations of our method.

### 5.1. Implementation details

**Model setup.** Our approach is implemented based on a publicly available repository<sup>2</sup>. In this implementation, a NeRF is parameterized by a multi-layer perception (MLP), with instant-npg [12] for positional encoding. To enhance photo-realism and enable flexible modeling of lighting, we discard the Lambertian shading as employed in [16]. Instead, we encode the ray direction using spherical harmonics and utilize it as an input to NeRF. Additionally, we incorporate a background network that predicts background color solely based on the ray direction. For our model, we

<sup>2</sup><https://github.com/ashawkey/stable-dreamfusion/tree/main>.

employ a pre-trained SD model<sup>3</sup> that takes input images at  $512 \times 512$  resolution. To predict the disparity map, we employ a pre-trained dense prediction transformer<sup>4</sup>.

**Training setup.** We use Adam [7] with a learning rate of  $10^{-2}$  for instant-npg encoding, and  $10^{-3}$  for NeRF weights. In practice, we choose total\_iter as  $10^4$  iterations. The rendering resolution is  $512 \times 512$ . To accelerate training, we employ DDIM [26] with empirically chosen parameters  $r = 0.25$ , and  $\eta = 1$ . We choose the hyper-parameters  $\lambda_{rgb} = 0.1$ ,  $\lambda_d = 0.1$ ,  $\lambda_{zvar} = 3$ . Similar to prior work [8, 16, 29], we use classifier-free guidance [4] of 100 for our diffusion model.

## 5.2. Experimental results

**Qualitative rendered results** of the 3D assets synthesized by our approach are depicted in Fig. 1. Our approach excels at generating high-fidelity 3D assets, producing photo-realistic and visually consistent images from multiple viewpoints.

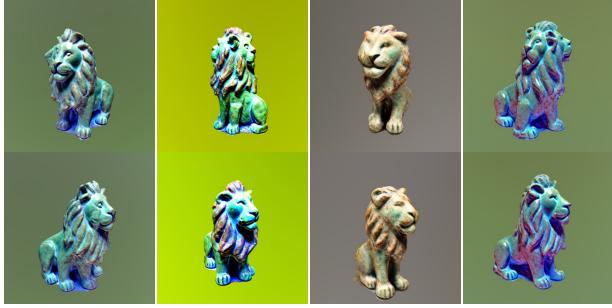
**Qualitative comparisons to state-of-the-art methods** are shown in Fig. 6 and Fig. 7. Specifically, as shown in Fig. 6, our rendered images exhibit enhanced photo-realism, showcasing improved texture details of the 3D assets and more plausible lighting effects. Similarly, when comparing to Score-Jacobian-Chaining [29] in Fig. 7, we observe a clear superiority in terms of vividness and the overall quality of image content.

**Ablation study** is conducted to validate the effectiveness of the proposed techniques, as illustrated in Fig. 9-10. Through these studies, we observe that the removal of each proposed technique can result in varying degrees of artifacts in RGB colors or 3D geometry. Moreover, in Fig. 10 we compare the rendered depth maps for images with either the full loss or with the loss  $\mathcal{L}_{zvar}$  removed. Our observation reveals that the removal of the loss  $\mathcal{L}_{zvar}$  results in both inaccurate geometry and a distorted rendered image.

**Limitations.** While our method successfully synthesizes plausible 3D assets, we have identified several limitations, as illustrated in Fig. 11, which we intend to address in future work. Firstly, we have observed instances where our model struggles to comprehend certain text prompts, as seen in the failures to synthesize "lox" and "peeling" depicted in Fig. 11. This limitation can be attributed to the constrained capacity of a pre-trained CLIP text encoder used in an SD model [22] for conditioning representation. We anticipate that employing a more advanced language model will alleviate this issue. Secondly, our approach yields unsatisfactory synthetic results in certain cases, such as the example of Zelda Link shown in Fig. 11 with 2 ears on one side and inaccurate facial reconstruction. To address this, it may be

<sup>3</sup>We use the public code base <https://github.com/huggingface/diffusers>.

<sup>4</sup><https://github.com/huggingface/transformers>.



$\mathcal{L}$        $\lambda_{\text{RGB}} = 0$        $w/o \text{ annealing}$        $w/o \mathcal{L}_d$

Figure 9. Ablation study for the proposed techniques. Specifically, we present 2-view (*row 1-2*) results in *column 1-5* with the full proposed method, the removal of  $\mathcal{L}_{\text{BGT}}$  in image space (i.e.,  $\lambda_{\text{RGB}} = 0$ ), annealed time step sampling (i.e., random sampling instead), and depth loss  $\mathcal{L}_d$ , respectively. Prompt: a ceramic lion.

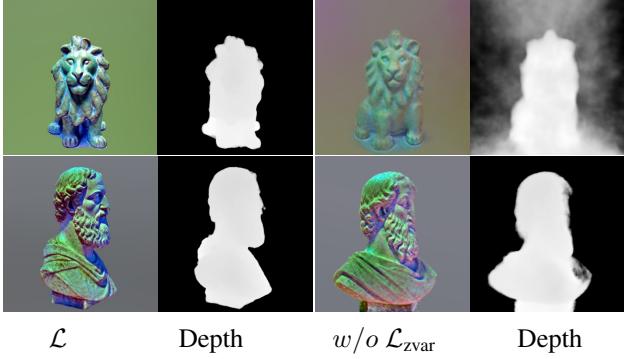


Figure 10. Ablation study for the loss  $\mathcal{L}_{\text{zvar}}$ . In this study, we showcase the 2-view (*row 1-2*) rendered images (*column 1 and 3*) along with their corresponding rendered depth image on their right (*column 2 and 4*). We observe that removal of the loss  $\mathcal{L}_{\text{zvar}}$  results in inaccurate geometry. Prompts: (1) a ceramic lion (*row 1*) (2) a highly detailed stone bust of Theodoros Kolokotronis (*row 2*).



Figure 11. Failure cases of our method include limited understanding of text prompts (*column 1-2*) and the presence of artifacts (*column 3-4*).

beneficial to enhance the capabilities of the 2D diffusion prior.

## 6. Conclusion

We propose a novel approach for text-to-3D synthesis. Our method leverages several key enhancements, including improved guidance from the diffusion prior, additional supervision on latent vectors and rendered images of the diffusion model, as well as depth and volume density of NeRFs. Extensive experiments highlight that our method significantly enhances the performance w.r.t. photo-realism and multi-view consistency, compared to state-of-the-art methods.

## References

- [1] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. *arXiv preprint arXiv:2303.13873*, 2023. [4](#), [5](#)
- [2] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022. [4](#)
- [3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020. [4](#), [7](#)
- [4] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. [9](#)
- [5] Ajay Jain, Ben Mildenhall, Jonathan T. Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. *CVPR*, 2022. [4](#)
- [6] Nasir Mohammad Khalid, Tianhao Xie, Eugene Belilovsky, and Popa Tiberiu. Clip-mesh: Generating textured meshes from text using pretrained image-text models. *SIGGRAPH Asia 2022 Conference Papers*, December 2022. [4](#)
- [7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015. [9](#)
- [8] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *CVPR*, 2023. [4](#), [5](#), [6](#), [9](#)
- [9] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *NeurIPS*, 2022. [4](#)
- [10] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2021. [4](#)
- [11] 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. *ECCV*, 2020. [4](#), [8](#)
- [12] 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. [9](#)

- [13] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *ICCV*, 2019. 4
- [14] Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 8
- [15] Junyi Pan, Xiaoguang Han, Weikai Chen, Jiapeng Tang, and Kui Jia. Deep mesh reconstruction from single rgb images via topology modification networks. In *ICCV*, 2019. 4
- [16] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 4, 5, 6, 7, 9
- [17] 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 *ICML*. PMLR, 2021. 4
- [18] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *NeurIPS*, 2022. 4
- [19] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, 2021. 4
- [20] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *CVPR*, 2021. 8
- [21] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *CVPR*, 2017. 4
- [22] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 4, 5, 6, 7, 8, 9
- [23] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS*, 2022. 4, 5
- [24] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *ICLR*, 2022. 4
- [25] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015. 4
- [26] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *ICLR*, 2021. 4, 7, 9
- [27] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *ICML*, 2023. 4
- [28] Junshu Tang, Tengfei Wang, Bo Zhang, Ting Zhang, Ran Yi, Lizhuang Ma, and Dong Chen. Make-it-3d: High-fidelity 3d creation from a single image with diffusion prior. *arXiv preprint arXiv:2303.14184*, 2023. 4, 5
- [29] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. *CVPR*, 2023. 4, 5, 6, 7, 9
- [30] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *ECCV*, 2018. 4
- [31] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T Freeman, and Joshua B Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *NeurIPS*, 2016. 4
- [32] Jiale Xu, Xiantao Wang, Weihao Cheng, Yan-Pei Cao, Ying Shan, Xiaohu Qie, and Shenghua Gao. Dream3d: Zero-shot text-to-3d synthesis using 3d shape prior and text-to-image diffusion models. *arXiv preprint arXiv:2212.14704*, 2022. 4
- [33] Peiye Zhuang, Samira Abnar, Jiatao Gu, Alex Schwing, Joshua M Susskind, and Miguel Ángel Bautista. Diffusion probabilistic fields. In *ICLR*, 2023. 4