

# Dream3D: Zero-Shot Text-to-3D Synthesis Using 3D Shape Prior and Text-to-Image Diffusion Models

Jiale Xu<sup>1,3\*</sup> Xintao Wang<sup>1†</sup> Weihao Cheng<sup>1</sup> Yan-Pei Cao<sup>1</sup>

Ying Shan<sup>1</sup> Xiaohu Qie<sup>2</sup> Shenghua Gao<sup>3,4,5†</sup>

<sup>1</sup>ARC Lab, <sup>2</sup>Tencent PCG <sup>3</sup>ShanghaiTech University

<sup>4</sup>Shanghai Engineering Research Center of Intelligent Vision and Imaging

<sup>5</sup>Shanghai Engineering Research Center of Energy Efficient and Custom AI IC

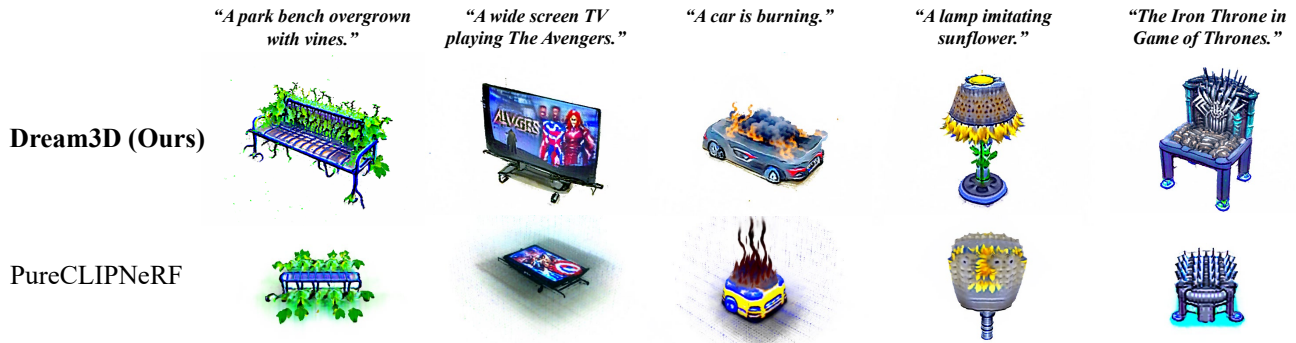


Figure 1. By utilizing 3D shape priors and powerful text-to-image diffusion models, our Dream3D can generate 3D content that exhibits superior visual quality and shape accuracy in accordance with the text prompt when compared to PureCLIPNeRF [25].

## Abstract

Recent CLIP-guided 3D optimization methods, such as DreamFields [20] and PureCLIPNeRF [25], have achieved impressive results in zero-shot text-to-3D synthesis. However, due to scratch training and random initialization without prior knowledge, these methods often fail to generate accurate and faithful 3D structures that conform to the input text. In this paper, we make the first attempt to introduce explicit 3D shape priors into the CLIP-guided 3D optimization process. Specifically, we first generate a high-quality 3D shape from the input text in the text-to-shape stage as a 3D shape prior. We then use it as the initialization of a neural radiance field and optimize it with the full prompt. To address the challenging text-to-shape generation task, we present a simple yet effective approach that directly bridges the text and image modalities with a powerful text-to-image diffusion model. To narrow the style domain gap between the images synthesized by the text-to-image diffusion model and shape renderings used to train the image-to-shape generator, we further propose to jointly optimize a learnable text prompt and fine-tune the text-to-image diffusion model for rendering-style image generation. Our method,

*Dream3D, is capable of generating imaginative 3D content with superior visual quality and shape accuracy compared to state-of-the-art methods. Our project page is at <https://bluestyle97.github.io/dream3d/>.*

## 1. Introduction

Text-to-3D synthesis endeavors to create 3D content that is coherent with an input text, which has the potential to benefit a wide range of applications such as animations, games, and virtual reality. Recently developed zero-shot text-to-image models [36, 46, 47, 50, 52] have made remarkable progress and can generate diverse, high-fidelity, and imaginative images from various text prompts. However, extending this success to the text-to-3D synthesis task is challenging because it is not practically feasible to collect a comprehensive paired text-3D dataset.

Zero-shot text-to-3D synthesis [20, 23, 25, 42, 53], which eliminates the need for paired data, is an attractive approach that typically relies on powerful vision-language models such as CLIP [62]. There are two main categories of this approach. 1) CLIP-based generative models, such as CLIP-Forge [53]. They utilize images as an intermediate bridge and train a mapper from the CLIP image embeddings of

\*Work done during an internship at ARC Lab, Tencent PCG.

†Corresponding Author.

ShapeNet renderings to the shape embeddings of a 3D shape generator, then switch to the CLIP text embedding as the input at test time. **2)** CLIP-guided 3D optimization methods, such as DreamFields [20] and PureCLIPNeRF [25]. They continuously optimize the CLIP similarity loss between a text prompt and rendered images of a 3D scene representation, such as neural radiance fields [1, 27, 34, 43]. While the first category heavily relies on 3D shape generators trained on limited 3D shapes and seldom has the capacity to adjust its shape structures, the second category has more creative freedom with the “dreaming ability” to generate diverse shape structures and textures.

We develop our method building upon CLIP-guided 3D optimization methods. Although these methods can produce remarkable outcomes, they typically fail to create precise and accurate 3D structures that conform to the input text (Fig. 1, 2<sup>nd</sup> row)). Due to the scratch training and random initialization without any prior knowledge, these methods tend to generate highly-unconstrained “adversarial contents” that have high CLIP scores but low visual quality. To address this issue and synthesize more faithful 3D contents, we suggest generating a high-quality 3D shape from the input text first and then using it as an explicit “**3D shape prior**” in the CLIP-guided 3D optimization process. In the *text-to-shape*\* stage, we begin by synthesizing a 3D shape without textures of the main common object in the text prompt. We then use it as the *initialization* of a voxel-based neural radiance field and optimize it with the full prompt.

The *text-to-shape* generation itself is a challenging task. Previous methods [20, 53] are often trained on images and tested with texts, and use CLIP to bridge the two modalities. However, this approach leads to a mismatching problem due to the gap between the CLIP text and image embedding spaces. Additionally, existing methods cannot produce high-quality 3D shapes. In this work, we propose to directly bridge the text and image modalities with a powerful text-to-image diffusion model, *i.e.*, Stable Diffusion [50]. We use the *text-to-image* diffusion model to synthesize an image from the input text and then feed the image into an *image-to-shape* generator to produce high-quality 3D shapes. Since we use the same procedure in both training and testing, the mismatching problem is largely reduced. However, there is still a style domain gap between the images synthesized by Stable Diffusion and the shape renderings used to train the image-to-shape generator. Inspired by recent work on controllable text-to-image synthesis [12, 51], we propose to jointly optimize a learnable text prompt and fine-tune the Stable Diffusion to address this domain gap. The fine-tuned Stable Diffusion can reliably synthesize images in the style of shape renderings used to train the image-

to-shape module without suffering from the domain gap.

To summarize, **1)** We make the first attempt to introduce the explicit **3D shape prior** into CLIP-guided 3D optimization methods. The proposed method can generate more accurate and high-quality **3D shapes** conforming to the corresponding text, while still enjoying the “**dreaming**” ability of generating diverse shape structures and textures (Fig. 1, 1<sup>st</sup> row). Therefore, we name our method “Dream3D” as it has both strengths. **2)** Regarding text-to-shape generation, we present a straightforward yet effective approach that directly connects the text and image modalities using a powerful text-to-image diffusion model. To narrow the style domain gap between the synthesized images and shape renderings, we further propose to jointly optimize a learnable text prompt and fine-tune the text-to-image diffusion model for rendering-style image generation. **3)** Our Dream3D can generate imaginative 3D content with better visual quality and shape accuracy than state-of-the-art methods. Additionally, our text-to-shape pipeline can produce 3D shapes of higher quality than previous work.

## 2. Related Work

**3D Shape Generation.** Generative models for 3D shapes have been extensively studied in recent years. It is more challenging than 2D image generation due to the expensive 3D data collection and the complexity of 3D shapes. Various 3D generators employ different shape representations, *e.g.*, voxel grids [3, 26, 59], point clouds [2, 65, 67, 69, 72], meshes [14, 15, 17, 35], and implicit fields [8, 30, 63, 64, 71]. These generators are trained to model the distribution of shape geometry (and optionally, texture) from a collection of 3D shapes. Some methods [4, 5, 13, 38, 40, 41, 55] attempt to learn a 3D generator using only 2D image supervision. These methods incorporate explicit 3D representations, such as meshes [13, 40, 41] and neural radiance fields [4, 5, 38, 55], along with surface or volume-based differentiable rendering techniques [21, 22, 34, 37], to enable the learning of 3D awareness from images.

**Text-to-Image.** Previous studies in text-to-image synthesis [44, 48, 66, 70] have focused mainly on domain-specific datasets and utilized GANs [16]. However, recent advances in scalable generative architectures and large-scale text-image datasets [54] have enabled unprecedented performance in zero-shot text-to-image synthesis. DALL-E [47] and GLIDE [36], as pioneering works, employ autoregressive model [11] and diffusion model [19, 57] as their architectures, respectively. DALL-E 2 [46] utilizes a diffusion prior network to translate CLIP text embeddings to CLIP image embeddings, and an unCLIP module to synthesize images from CLIP image embeddings. In Imagen [52] and Stable Diffusion [50], a large pre-trained text encoder is employed to guide the sampling process of a diffusion model in pixel space and latent space, respectively.

\*Throughout this paper, we use the term “shape” to refer to 3D geometric models *without textures*, while some works [7, 29] also use this term for textured 3D models.

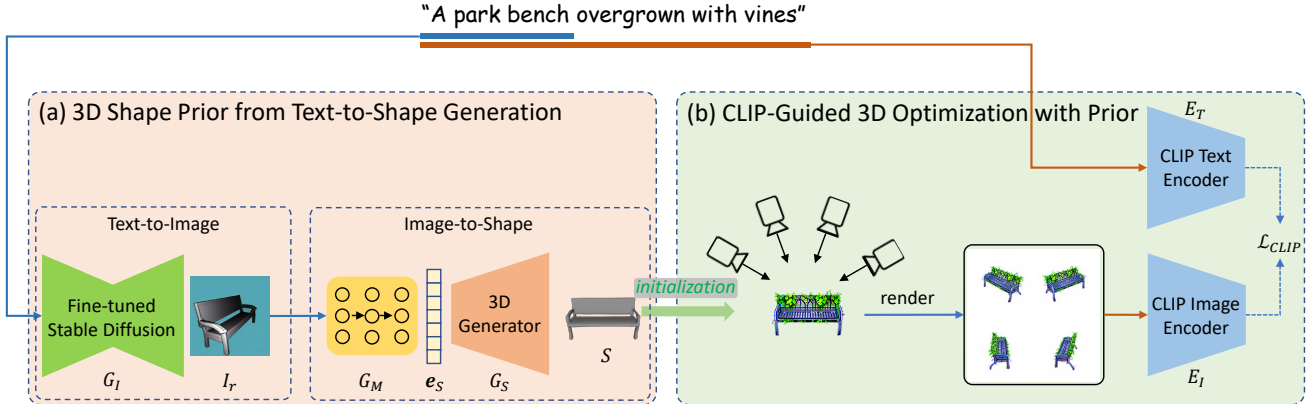


Figure 2. Overview of our text-to-3D synthesis framework. (a) In the first text-to-shape stage, a fine-tuned Stable Diffusion  $G_I$  is employed to synthesize a rendering-style image  $I_r$  from the input text prompt  $y$ . This image is then used to generate a latent shape embedding  $e_S$  with the assistance of a shape embedding generation network  $G_M$ . Finally, the high-quality 3D shape generator  $G_S$  leverages  $e_S$  to produce a 3D shape  $S$ , which is used as an explicit 3D shape prior. (b) In the second optimization stage, the 3D shape prior  $S$  is utilized to initialize a neural radiance field, which is further optimized with CLIP guidance to synthesize 3D content that is consistent with the input text prompt  $y$ .

**Zero-Shot Text-to-3D.** Zero-shot text-to-3D generation techniques [20, 23, 25, 33, 42, 53] exploit the joint text-image modeling capability of pre-trained vision-language models such as CLIP [45] to obviate the need for paired text-3D data. CLIP-Forge [53] trains a normalizing flow [10, 49] model to convert CLIP image embeddings to VAE [24] shape embeddings, and switches the input to CLIP text embeddings at the inference time. ISS [28] trains a mapper to map the CLIP image embedding into the latent shape code of a pre-trained single-view reconstruction (SVR) network [37], which is then fine-tuned by taking the CLIP text embedding as input. DreamFields [20] and CLIP-Mesh [23] are pioneering works that explore zero-shot 3D content creation using only CLIP guidance. The former optimizes a randomly-initialized NeRF, while the latter optimizes a spherical template mesh as well as random texture and normal maps. PureCLIPNeRF [25] enhances DreamFields with grid-based representation [58] and more diverse image augmentations. Recently, DreamFusion [42] has gained popularity in the research community due to its impressive results. Powered by a strong text-to-image model, Imagen [52], it can generate high-fidelity 3D objects using the score distillation loss.

### 3. Method

Our objective is to generate 3D content that aligns with the given input text prompt  $y$ . As illustrated in Fig. 2, our framework for text-guided 3D synthesis comprises two stages. In the first stage (Sec. 3.2), we obtain an explicit 3D shape prior  $S$  using a text-guided 3D shape generation process. The text-guided shape generation process involves a text-to-image phase that employs a fine-tuned Stable Diffusion model [50]  $G_I$  (Sec. 3.3), and an image-to-shape phase

that employs a shape embedding generation network  $G_M$  and a high-quality 3D shape generator  $G_S$ . In the second stage (Sec. 3.1), we utilize the 3D prior  $S$  to initialize a neural radiance field [34], and optimize it with CLIP [45] guidance to generate the 3D content. Our framework only requires a collection of textureless 3D shapes without any text labels to train the 3D generator  $G_S$ , and the fine-tuning process of  $G_I$  converges rapidly.

#### 3.1. CLIP-Guided 3D Optimization with explicit 3D Shape Prior

**Background: 3D Optimization with CLIP Guidance.** CLIP [45] is a powerful vision-language model that comprises a text encoder  $E_T$ , and an image encoder  $E_I$ . By maximizing the cosine similarity between the text embedding and the image embedding encoded by  $E_T$  and  $E_I$  respectively on a large-scale paired text-image dataset, CLIP aligns the text and image modalities in a shared latent embedding space.

Prior research [20, 23, 25] leverages the capability of CLIP [45] to generate 3D contents from text. Starting from a randomly-initialized 3D representation parameterized by  $\theta$ , they render images from multiple viewpoints and optimize  $\theta$  by minimizing the CLIP similarity loss between the rendered image  $\mathcal{R}(v_i; \theta)$  and the text prompt  $y$ :

$$\mathcal{L}_{\text{CLIP}} = -E_I(\mathcal{R}(\theta; v_i))^T E_T(y), \quad (1)$$

where  $\mathcal{R}$  denotes the rendering process, and  $v_i$  denotes the rendering viewpoint at the  $i$ -th optimization step. Specifically, DreamFields [20] and PureCLIPNeRF [25] employ neural radiance fields [34] (NeRF) as the 3D representation  $\theta$ , while CLIP-Mesh [23] uses a spherical template mesh with associated texture and normal maps.

**Observations and Motivations.** Though these CLIP-guided optimization methods can generate impressive results, we observe that they often fall short in producing precise and detailed 3D structures that accurately match the text description. As depicted in Fig. 1, we employ these methods to create 3D content featuring common objects, but the outcomes exhibited distortion artifacts and appeared unusual, adversely affecting their visual quality and hindering their use in real-world applications.

We attribute the failure of previous works to generate accurate and realistic objects to two main factors: (i) The optimization process begins with a *randomly-initialized 3D representation* lacking any explicit 3D shape prior, making it very challenging for the models to conjure up the scene from scratch. (ii) The CLIP loss in Eq. (1) prioritizes global consistency between the rendered image and the text prompt, rather than offering robust and precise guidance on the synthesized 3D structure. As a result, the optimization output is significantly unconstrained.

**Optimization with 3D Shape Prior as Initialization.** To address the aforementioned issue and generate more faithful 3D content, we propose to use a text-to-shape generation process to create a *high-quality 3D shape*  $S$  from the input text prompt  $y$ . Subsequently, we use it as an explicit “3D shape prior” to initialize the CLIP-guided 3D optimization process. As illustrated in Fig. 2, for the text prompt “*a park bench overgrown with vines*”, we first synthesize “*a park bench*” without textures in the text-to-shape stage. We then use it as the initialization of a neural radiance field and optimize it with the full prompt, following previous works.

Our optimization process utilizes DVGO [58], an efficient NeRF variant that represents NeRF using a density voxel grid  $\mathbf{V}_{\text{density}} \in \mathbb{R}^{N_x \times N_y \times N_z}$  and a shallow color MLP  $f_{\text{rgb}}$ . We start by taking a 3D shape  $S$  generated by the text-to-shape process, represented as an SDF grid  $\tilde{\mathbf{V}}_{\text{sdf}} \in \mathbb{R}^{N_x \times N_y \times N_z}$ , and transform it into the density voxel grid  $\mathbf{V}_{\text{density}}$  using the following equations [39, 58, 68]:

$$\Sigma = \frac{1}{\beta} \text{sigmoid} \left( -\frac{\tilde{\mathbf{V}}_{\text{sdf}}}{\beta} \right), \quad (2a)$$

$$\mathbf{V}_{\text{density}} = \max(0, \text{softplus}^{-1}(\Sigma)) \quad (2b)$$

Here,  $\text{sigmoid}(x) = 1/(1 + e^{-x})$  and  $\text{softplus}^{-1}(x) = \log(e^x - 1)$ . Eq. (2a) converts SDF values to density for volume rendering, where  $\beta > 0$  is a hyper-parameter controlling the sharpness of the shape boundary (smaller  $\beta$  leads to sharper shape boundary,  $\beta = 0.05$  in our experiments). Eq. (2b) transforms the density into pre-activated density. To ensure that the distribution of the accumulated transmittance is the same as DVGO, we clamp the minimum value of the density outside the shape prior as 0.

With the density grid  $\mathbf{V}_{\text{density}}$  initialized by the 3D shape  $S$  and the color MLP  $f_{\text{rgb}}$  initialized randomly, we render

image  $\mathcal{R}(\mathbf{V}_{\text{density}}, f_{\text{rgb}}; \mathbf{v}_i)$  from viewpoint  $\mathbf{v}_i$  and optimize  $\theta = (\mathbf{V}_{\text{density}}, f_{\text{rgb}})$  with the CLIP loss in Eq. (1). Following DreamFields [20] and PureCLIPNeRF [25], we perform background augmentations for the rendered images and leverage the transmittance loss introduced by [20] to reduce noise and spurious density. Besides, since CLIP loss cannot provide accurate geometrical supervision, the 3D shape prior may be gradually disturbed and “forgotten”, thus we also adopt a shape-prior-preserving loss to preserve the global structure of the 3D shape prior:

$$\mathcal{L}_{\text{prior}} = - \sum_{(x,y,z)} \mathbb{1}(\tilde{\mathbf{V}}_{\text{sdf}} < 0) \cdot \text{alpha}(\mathbf{V}_{\text{density}}), \quad (3)$$

where  $\mathbb{1}(\cdot)$  is the indicator function, and  $\text{alpha}(\cdot)$  transforms the density into the opacity representing the probability of termination at each position in volume rendering.

By initializing NeRF with an explicit 3D shape prior, we give extra knowledge on how the 3D content should look like and prevent the model from imagining from scratch and generating “adversarial contents” that have high CLIP scores but low visual quality. Based on the initialization, the CLIP-guided optimization further provides flexibility and is able to synthesize *more diverse structures and textures*.

### 3.2. Stable-Diffusion-Assisted Text-to-Shape Generation as 3D Shape Prior

To obtain the 3D shape prior, a text-guided shape generation scheme is required, which is a challenging task due to the lack of paired text-shape datasets. Previous approaches [28, 53] typically first train an image-to-shape model using *rendered* images, and then bridge the text and image modalities using the CLIP embedding space.

CLIP-Forge [53] trains a normalizing flow network to map CLIP image embeddings of shape renderings to latent embeddings of a volumetric shape auto-encoder, and at test time, it switches to CLIP text embeddings as input. However, the shape auto-encoder has difficulty in generating high-quality and diverse 3D shapes, and directly feeding CLIP text embeddings to the flow network trained on CLIP image embeddings suffers from the gap between the CLIP text and image embedding spaces. ISS [28] trains a mapper network to map CLIP image embeddings of shape renderings to the latent space of a pre-trained single-view reconstruction (SVR) model, and fine-tunes the mapper at test time by maximizing the CLIP similarity between the text prompt and the images rendered from synthesized shapes. While the test-time fine-tuning alleviates the gap between the CLIP text and image embeddings, it is cumbersome to fine-tune the mapper for each text prompt.

In contrast to the aforementioned methods that connect the text and image modalities in the CLIP embedding space, we use a powerful text-to-image diffusion model to directly bridge the two modalities. Specifically, we first synthesize

an image from the input text and then feed it into an image-to-shape module to generate a *high-quality* 3D shape. This pipeline is more concise and naturally eliminates the gap between CLIP text and image embeddings. However, it introduces a new domain gap between the images generated by the text-to-image diffusion model and the shape renderings used to train the image-to-shape module. We will introduce a novel technique to alleviate this gap in Sec. 3.3.

**Text-to-Image Diffusion Model.** Diffusion models [19,57] are generative models trained to reverse a diffusion process. The diffusion process begins with a sample from the data distribution,  $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ , which is gradually corrupted by Gaussian noise over  $T$  timesteps:  $\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\epsilon_{t-1}$ ,  $t = 1, 2, \dots, T$ , where  $\alpha_t$  defines the noise level and  $\epsilon_{t-1}$  denotes the noise added at timestep  $t-1$ . To reverse this process, a denoising network  $\epsilon_\theta$  is trained to estimate the added noise at each timestep. During inference, samples can be generated by iteratively denoising pure Gaussian noise. Text-to-image diffusion models further condition the denoising process on texts. Given a text prompt  $y$  and a text encoder  $c_\theta$ , the training objective is:

$$\mathcal{L}_{\text{diffusion}} = \mathbb{E}_{\mathbf{x}_0, t, \epsilon_t, y} \|\epsilon_t - \epsilon_\theta(\mathbf{x}_t, t, c_\theta(y))\|_2^2 \quad (4)$$

In this work, we use Stable Diffusion<sup>†</sup> [50], an open-source text-to-image diffusion model, which employs a CLIP ViT-L/14 text encoder as  $c_\theta$  and is trained on the large-scale LAION-5B dataset [54]. Stable Diffusion is known for its ability to generate diverse and imaginative images in various styles from heterogeneous text prompts.

**High-quality 3D generator.** To provide a more precise initialization for optimization, high-quality 3D shapes are highly desirable. In this work, we utilize the SDF-StyleGAN [71], a state-of-the-art 3D generative model, to generate high-quality 3D priors. SDF-StyleGAN is a StyleGAN2-like architecture that maps a random noise  $z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  to a latent shape embedding  $e_S \in \mathcal{W}$  and synthesizes a 3D feature volume  $F_V$ , which is an implicit shape representation. We can query the SDF value at arbitrary position  $\mathbf{x}$  by feeding the interpolated feature from  $F_V$  at  $\mathbf{x}$  into a jointly trained MLP. We improve upon the original SDF-StyleGAN, which trains one network for each shape category, by training a *single* 3D shape generator  $G_S$  on the 13 categories of ShapeNet [6]. This modification provides greater flexibility in the text-to-shape process.

**Shape Embedding Mapping Network.** To bridge the image and shape modalities, we further train a shape embedding mapping network  $G_M$ . Firstly, we utilize  $G_S$  to generate a large set of 3D shapes  $\{S^i\}_{i=1}^N$  and shape embeddings  $\{e_S^i\}_{i=1}^N$ . Then, we render  $S^i$  from  $K$  viewpoints to obtain shape renderings  $\{I_r^j\}_{j=1}^{NK}$  and the corresponding image embeddings  $\{e_I^j\}_{j=1}^{NK}$  with the CLIP image encoder  $E_I$ , forming a paired image-shape embedding dataset

<sup>†</sup><https://github.com/CompVis/stable-diffusion>

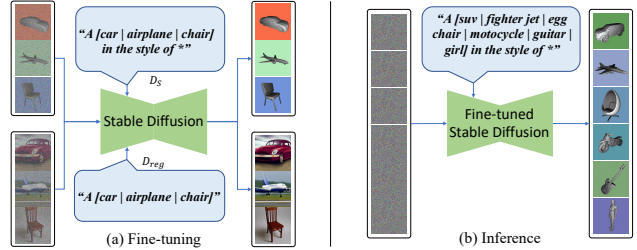


Figure 3. Fine-tuning Stable Diffusion into a stylized generator. (a) We fine-tune the text embedding  $v_*$  of a placeholder token  $*$  and the weights of Stable Diffusion using a dataset  $D_S$  that contains ShapeNet renderings and text descriptions in the format of “a CLS in the style of  $*$ ”. Additionally, we use another dataset  $D_{reg}$ , which contains images synthesized by the original Stable Diffusion with text prompts in the format of “a CLS”, for regularization purposes. (b) With the fine-tuned Stable Diffusion, we can synthesize images that match the style of ShapeNet renderings by appending the postfix “in the style of  $*$ ” to the text prompt.

$\{(e_I^j, e_S^i)\}_{j=1}^{NK}$ . Finally, we use this dataset to train a conditional diffusion model  $G_M$  which can synthesize shape embeddings from image embeddings of shape renderings.

To prepare the dataset for training  $G_M$ , we generate  $N = 64000$  shapes using  $G_S$  and render  $K = 24$  views for each shape. The ranges of azimuth and elevation angles of the rendered views are  $[-90^\circ, 90^\circ]$  and  $[20^\circ, 30^\circ]$ , respectively. We employ an SDF renderer [21] since we represent the synthesized shapes with SDF grids.

### 3.3. Fine-tuning Stable Diffusion for Rendering-Style Image Generation

As stated in Sec. 3.2, we use Stable Diffusion to directly bridge the text and image modalities for text-to-shape generation. Nonetheless, the image-to-shape module is trained on shape renderings, which exhibit a significant style domain gap from the images produced by Stable Diffusion. Previous research [28,53] has attempted to combine a text-to-image model with an image-to-shape model for text-to-shape generation. However, this approach is plagued by the aforementioned style domain gap, leading to flawed geometric structures and diminished performance.

Inspired by recent work on controllable text-to-image generation such as textual inversion [12] and Dream-Booth [51], we propose a method for addressing the domain gap problem by fine-tuning Stable Diffusion into a stylized generator. Our core idea is to enable Stable Diffusion to replicate the style of shape renderings used to train the image-to-shape module outlined in Sec. 3.2. This allows us to seamlessly input the generated *stylized* images into the image-to-shape module without being affected by the domain gap.

**Fine-tuning Process.** The fine-tuning process is illustrated in Fig. 3. To fine-tune Stable Diffusion, we need a dataset



Figure 4. Qualitative comparison on text-guided 3D synthesis. The 3D shape prior used to initialize the CLIP-guided optimization process for each of our results (the last row) is also visualized below.

that consists of shape renderings and related *stylized* text prompts. For each shape  $S$  in the ShapeNet dataset, we generate a set of shape renderings  $\{I_S^j\}_{j=1}^{N_S}$ . Subsequently, each rendering  $I_S^j$  is linked with a *stylized* text prompt  $y_S^j$  in the format of “*a CLS in the style of \**”, where *CLS* denotes the shape category name and  $*$  represents a placeholder token that requires optimization for its text embedding. For instance, if the image is rendered from a chair shape, then the associated text prompt will be “*a chair in the style of \**”. The paired dataset  $D_S = (I_S^j, y_S^j)_{j=1}^{N_S}$  is then utilized to fine-tune Stable Diffusion by minimizing  $\mathcal{L}_{\text{diffusion}}$  presented in Eq. (4).

During fine-tuning, we freeze the CLIP text encoder of Stable Diffusion, and optimize two objectives: (i) the text embedding of the placeholder token  $*$ , denoted as  $v_*$ , and (ii) the parameters  $\theta$  of the diffusion model  $\epsilon_\theta$ . Optimizing the text embedding  $v_*$  aims to learn a virtual word that captures the style of the rendered images best, even though it is not present in the vocabulary of the text encoder. Fine-tuning the parameters  $\theta$  of the diffusion model further enhances the ability to capture the style precisely since it is hard to control the synthesis of Stable Diffusion solely on the language level. Our experiments demonstrate stable convergence of the fine-tuning process in approximately 2000 optimization steps, requiring only 40 minutes on a single Tesla A100 GPU. We show some synthesized results using the fine-tuned model in Fig. 3.

**Dataset Scale and Background Augmentation.** We have identified two essential techniques empirically that enable the fine-tuned model to synthesize stylized images in a stable manner. Firstly, unlike textual inversion [12] or Dream-

Booth [51] which utilize only 3–5 images, fine-tuning with a larger set of shape renderings containing thousands of images helps the model capture the style more precisely. Secondly, fine-tuning Stable Diffusion using shape renderings with a pure-white background results in a chaotic and uncontrollable background during inference. However, augmenting the shape renderings with random solid-color backgrounds allows the fine-tuned model to synthesize images with solid-color backgrounds stably, making it easy to remove the background if necessary. Further details can be found in the supplementary material.

## 4. Experiments

In this section, we evaluate the efficacy of our proposed text-to-3D synthesis framework. Initially, we compare our results with state-of-the-art techniques (Sec. 4.1). Subsequently, we demonstrate the effectiveness of our Stable-Diffusion-assisted approach for text-to-shape generation (Sec. 4.2). Furthermore, we conduct ablation studies to assess the effectiveness of critical components of our framework (Sec. 4.3).

**Dataset.** Our framework requires only a set of untextured 3D shapes for training the 3D shape generator  $G_S$ . Specifically, we employ 13 categories from ShapeNet [6] and utilize the data preprocessing procedure of Zheng *et al.* [71] to generate  $128^3$  SDF grids from the original meshes. During the fine-tuning of Stable Diffusion, we employ the SDF renderer [21] to produce a shape rendering dataset.

**Implementation details.** The 3D generator  $G_S$  utilizes the SDF-StyleGAN architecture. The diffusion-model-based shape embedding mapping network  $G_M$  is based on an

Method	CLIP R-Precision $\uparrow$	
	ViT-B/32	ViT-B/16
DreamFields [20]	63.24	92.65
CLIP-Mesh [23]	75.00	91.18
PureCLIPNeRF [25]	73.53	88.24
Ours w/o 3D prior	75.47	94.34
Ours	<b>85.29</b>	<b>98.53</b>

Table 1. Quantitative comparison on Text-guided 3D synthesis. All methods employ CLIP ViT/16 as the guiding model for optimization, while two distinct CLIP models are utilized to compute the CLIP retrieval precision.

open-source DALL-E 2 implementation<sup>‡</sup>. We train  $G_M$  by extracting image embeddings from shape renderings using the CLIP ViT-B/32 image encoder. The stylized text-to-image generator  $G_I$  is fine-tuned from Stable Diffusion v1.4. In the optimization stage, we set the learning rates for the density grid  $V_{\text{density}}$  and color MLP  $f_{\text{rgb}}$  to  $5 \times 10^{-1}$  and  $5 \times 10^{-3}$  respectively, and we adopt the CLIP ViT-B/16 encoder as the guidance model. For each text prompt, we optimize for 5000 steps, while previous NeRF-based text-to-3D methods [20, 25] typically require 10000 steps or more.

**Evaluation Metrics.** Regarding the primary results of our framework, *i.e.*, text-guided 3D content synthesis, we report the CLIP retrieval precision on a manually created dataset of diverse text prompts and objects. For specifics regarding the dataset, please refer to the supplementary material. This metric quantifies the percentage of generated images that the CLIP encoder associates with the correct text prompt used for generation. We utilize Fréchet Inception Distance (FID) [18] to evaluate the shape generation quality for the initial text-to-shape generation stage.

### 4.1. Text-Guided 3D Synthesis

We compare our method with three state-of-the-art baseline methods on the task of text-guided 3D synthesis, *i.e.*, DreamFields [20], CLIP-Mesh [23], and PureCLIPNeRF [25]. We conduct tests using the default settings and official implementations for all baseline methods. In particular, we utilize the medium-quality configuration of DreamFields and the implicit architecture variant of PureCLIPNeRF owing to its superior performance.

**Time cost.** Thanks to the shape prior initialization, our CLIP-guided optimization process exhibits significantly improved efficiency compared to previous NeRF-based methods. Dream3D optimizes for only 5000 steps within 25 minutes, while DreamFields and PureCLIPNeRF require more than 10000 steps, taking over an hour (measured on 1 A100 GPU). Training the 3D shape generator  $G_S$  takes 7 days on 4 A100 GPUs and training the shape embedding generation network  $G_M$  takes 1 day on 1 A100 GPU. It is worth noting that these models are only trained once and their inference

<sup>‡</sup><https://github.com/lucidrains/DALLE2-pytorch>

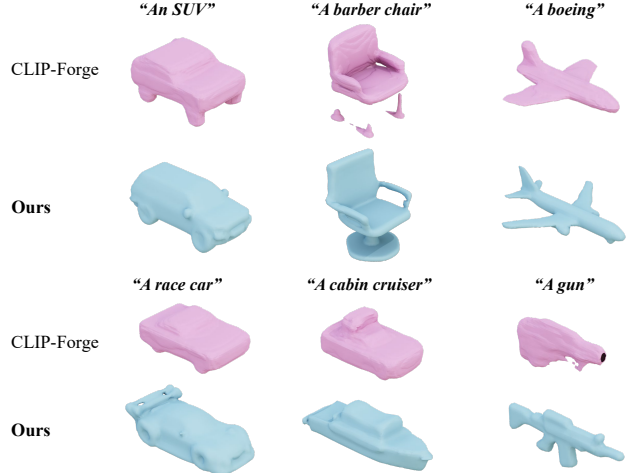


Figure 5. Text-guided 3D shape generation results. All the visualized meshes are extracted at the resolution of  $64^3$ . We can observe that our method can generate significantly more plausible 3D shapes benefiting from the high-quality 3D shape generator.

Method	FID $\downarrow$
CLIP-Forge [53]	112.38
Ours w/o text-to-image	58.36
Ours w/o fine-tuning SD	61.88
Ours	<b>40.83</b>

Table 2. Quantitative comparison on text-to-shape generation and ablation studies on the efficacy of fine-tuning Stable Diffusion.

time can be neglected compared to the optimization cost.

**Quantitative Results.** We report the CLIP retrieval precision metrics in Tab. 1. It is noteworthy that both the baseline methods and our approach utilize the CLIP ViT-B/16 encoder for optimization, and both the CLIP ViT-B/16 and CLIP ViT-B/32 encoders are employed as retrieval models. As Table Tab. 1 shows, our method achieves the highest CLIP R-Precision with both retrieval models. Moreover, our framework exhibits a significantly smaller performance gap between the two retrieval models compared to the baseline methods. By leveraging the 3D shape prior, our method initiates the optimization from a superior starting point, thereby mitigating the adversarial generation problem that prioritizes obtaining high CLIP scores while neglecting the visual quality. Consequently, our method demonstrates more robust performance across different CLIP models.

**Qualitative Results.** The qualitative comparison is presented in Fig. 4, indicating that the baseline methods [20, 23, 25] encounter challenges in generating precise and realistic 3D objects, resulting in distorted and unrealistic visuals. DreamFields’ results are frequently blurry and diffuse, while PureCLIPNeRF tends to synthesize symmetric objects. CLIP-Mesh experiences difficulties in generating intricate visual effects due to its explicit mesh representation. In contrast, our method effectively generates higher-quality 3D structures by incorporating explicit 3D shape priors.

## 4.2. Text-to-Shape Generation

The research on zero-shot text-to-shape generation is limited, and we compare our approach with CLIP-Forge [53] and measure the quality of shape generation using the Fréchet Inception Distance (FID). Specifically, we synthesize 3 shapes for each prompt from a dataset of 233 text prompts provided by CLIP-Forge. Then, we render 5 images for each synthesized shape and compare them to a set of ground truth ShapeNet renderings to compute the FID. The ground truth images are obtained by randomly choosing 200 shapes from the test set of each ShapeNet category and rendering 5 views for each shape.

As shown in Tab. 2, our approach achieves a lower FID than CLIP-Forge. CLIP-Forge employs a volumetric shape auto-encoder to generate 3D shapes. However, the qualitative results in Fig. 5 indicate poor shape generation capability, making it difficult to generate plausible 3D shapes. A high-quality 3D shape prior is also advantageous for the optimization process as an excellent initialization.

## 4.3. Ablation Studies

**Effectiveness of Fine-tuning Stable Diffusion.** Our approach employs a fine-tuned Stable Diffusion to establish a connection between the text and image modalities. To evaluate its efficacy, we employ the original Stable Diffusion model to produce images from the text prompts of CLIP-Forge [53]. Subsequently, we employ these images to generate 3D shapes using the image-to-shape module. The results in the 3<sup>rd</sup> row of Tab. 2 indicate a decline in FID performance, suggesting that this approach would negatively impact the shape generation process. Furthermore, we directly test using text embedding to generate shape embeddings with  $G_M$ , which also leads to a decline in performance as seen in the 2<sup>nd</sup> row of Tab. 2.

**Effectiveness of 3D Shape Prior.** To validate the efficacy of the 3D shape prior, we eliminate the first stage of our framework and optimize from scratch using the same text prompts as presented in Sec. 4.1. Subsequently, we evaluate the CLIP retrieval precision. The outcomes presented in Tab. 1 indicate that optimizing without 3D shape prior results in a considerable decline in performance, thereby demonstrating its effectiveness.

**Effectiveness of  $\mathcal{L}_{\text{prior}}$ .** The 3D prior preserving loss  $\mathcal{L}_{\text{prior}}$  shown in Eq. (3) aims to reinforce the 3D prior during the optimization process in case that the prior is gradually disturbed and discarded. To demonstrate its effectiveness, we synthesized “a park bench” as a prior for the prompt “A park bench overgrown with vines”, and then optimize with and without  $\mathcal{L}_{\text{prior}}$  and compared the results, which are visualized in Fig. 6. The results indicate that using  $\mathcal{L}_{\text{prior}}$  during optimization helps maintain the structure of the 3D prior shape, while discarding it causes distortion and discontinuity artifacts, thereby disturbing the initial shape.

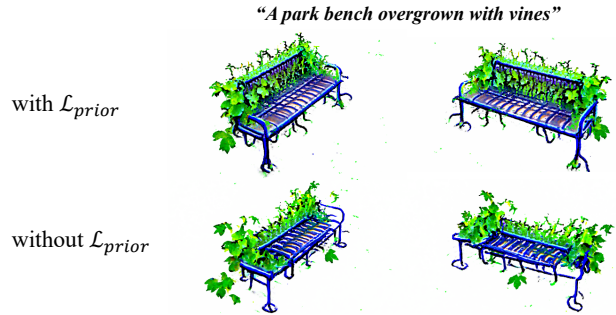


Figure 6. The effect of the 3D prior preserving loss  $\mathcal{L}_{\text{prior}}$ .

## 5. Limitations and Future Work

Our framework relies on a fine-tuned Stable Diffusion to generate rendering-style images. Despite its strong generation capability, Stable Diffusion may produce shape images that fall outside the distribution of the training data of the image-to-shape module. This is due to the fact that Stable Diffusion is trained on an internet-scale text-image dataset, whereas the 3D shape generator is trained on ShapeNet. Furthermore, the quality of text-to-shape synthesis in our framework is heavily reliant on the generation capability of the 3D generator. Our future work will explore incorporating stronger 3D priors into our framework to enable it to work with a wider range of object categories.

Additionally, our framework is indeed orthogonal to score distillation-based text-to-3D methods [32, 42, 60], as we can also utilize the score distillation sampling objective for optimization. We believe that incorporating 3D shape priors can enhance the quality and diversity of the generation results, as DreamFusion [42] acknowledged.

## 6. Conclusion

This paper introduces Dream3D, a text-to-3D synthesis framework that can generate diverse and imaginative 3D content from text prompts. Our approach incorporates explicit 3D shape priors into the CLIP-guided optimization process to generate more plausible 3D structures. To address the text-to-shape generation, we propose a straightforward yet effective method that utilizes a fine-tuned text-to-image diffusion model to bridge the text and image modalities. Our method is shown to generate 3D content with superior visual quality and shape accuracy compared to previous work, as demonstrated by extensive experiments.

**Acknowledgements.** The work was supported by National Key R&D Program of China (2018AAA0100704), NSFC #61932020, #62172279, Science and Technology Commission of Shanghai Municipality (Grant No. 20ZR1436000), Program of Shanghai Academic Research Leader, and “Shuguang Program” supported by Shanghai Education Development Foundation and Shanghai Municipal Education Commission.



## References

- [1] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5460–5469, 2022. [2](#)
- [2] Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. Learning gradient fields for shape generation. In *European Conference on Computer Vision*, pages 364–381. Springer, 2020. [2](#)
- [3] Yan-Pei Cao, Zheng-Ning Liu, Zheng-Fei Kuang, Leif Kobbelt, and Shi-Min Hu. Learning to reconstruct high-quality 3d shapes with cascaded fully convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 616–633, 2018. [2](#)
- [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, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16123–16133, June 2022. [2](#)
- [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 (CVPR)*, pages 5799–5809, June 2021. [2](#)
- [6] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. [5](#), [6](#), [12](#)
- [7] Kevin Chen, Christopher B Choy, Manolis Savva, Angel X Chang, Thomas Funkhouser, and Silvio Savarese. Text2shape: Generating shapes from natural language by learning joint embeddings. In *Asian conference on computer vision*, pages 100–116. Springer, 2018. [2](#)
- [8] 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 (CVPR)*, June 2019. [2](#)
- [9] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. [18](#)
- [10] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *International Conference on Learning Representations*, 2017. [3](#)
- [11] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12873–12883, June 2021. [2](#)
- [12] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022. [2](#), [5](#), [6](#)
- [13] 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. In *Advances In Neural Information Processing Systems*, 2022. [2](#)
- [14] Lin Gao, Tong Wu, Yu-Jie Yuan, Ming-Xian Lin, Yu-Kun Lai, and Hao Zhang. Tm-net: Deep generative networks for textured meshes. *ACM Transactions on Graphics (TOG)*, 40(6):263:1–263:15, 2021. [2](#)
- [15] Lin Gao, Jie Yang, Tong Wu, Yu-Jie Yuan, Hongbo Fu, Yu-Kun Lai, and Hao Zhang. Sdm-net: Deep generative network for structured deformable mesh. *ACM Transactions on Graphics (TOG)*, 38(6):1–15, 2019. [2](#)
- [16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. [2](#)
- [17] Kunal Gupta and Manmohan Chandraker. Neural mesh flow: 3d manifold mesh generation via diffeomorphic flows. In *Advances in Neural Information Processing Systems*, volume 33, pages 1747–1758, 2020. [2](#)
- [18] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. [7](#), [14](#)
- [19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. [2](#), [5](#)
- [20] Ajay Jain, Ben Mildenhall, Jonathan T. Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 867–876, June 2022. [1](#), [2](#), [3](#), [4](#), [7](#), [14](#)
- [21] Yue Jiang, Dantong Ji, Zhizhong Han, and Matthias Zwicker. Sdfdiff: Differentiable rendering of signed distance fields for 3d shape optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. [2](#), [5](#), [6](#)
- [22] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. [2](#)
- [23] Nasir Mohammad Khalid, Tianhao Xie, Eugene Belilovsky, and Popa Tiberiu. Clip-mesh: Generating textured meshes from text using pretrained image-text models. *SIGGRAPH Asia 2022 Conference Papers*, December 2022. [1](#), [3](#), [7](#), [14](#)
- [24] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. [3](#)
- [25] Han-Hung Lee and Angel X Chang. Understanding pure clip guidance for voxel grid nerf models. *arXiv preprint arXiv:2209.15172*, 2022. [1](#), [2](#), [3](#), [4](#), [7](#), [14](#)

- [26] Jun Li, Kai Xu, Siddhartha Chaudhuri, Ersin Yumer, Hao Zhang, and Leonidas Guibas. Grass: Generative recursive autoencoders for shape structures. *ACM Transactions on Graphics (TOG)*, 36(4):1–14, 2017. [2](#)
- [27] Jia-Wei Liu, Yan-Pei Cao, Weijia Mao, Wenqiao Zhang, David Junhao Zhang, Jussi Keppo, Ying Shan, Xiaoju Qie, and Mike Zheng Shou. Devrf: Fast deformable voxel radiance fields for dynamic scenes. *arXiv preprint arXiv:2205.15723*, 2022. [2](#)
- [28] Zhengzhe Liu, Peng Dai, Ruihui Li, Xiaojuan Qi, and Chi-Wing Fu. Iss: Image as stetting stone for text-guided 3d shape generation. *arXiv preprint arXiv:2209.04145*, 2022. [3](#), [4](#), [5](#), [18](#)
- [29] Zhengzhe Liu, Yi Wang, Xiaojuan Qi, and Chi-Wing Fu. Towards implicit text-guided 3d shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17896–17906, June 2022. [2](#)
- [30] Andrew Luo, Tianqin Li, Wen-Hao Zhang, and Tai Sing Lee. Surfgen: Adversarial 3d shape synthesis with explicit surface discriminators. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16238–16248, October 2021. [2](#)
- [31] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. [18](#)
- [32] Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. *arXiv preprint arXiv:2211.07600*, 2022. [8](#)
- [33] Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2mesh: Text-driven neural stylization for meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13492–13502, June 2022. [3](#)
- [34] 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. [2](#), [3](#), [13](#)
- [35] Charlie Nash, Yaroslav Ganin, S. M. Ali Eslami, and Peter W. Battaglia. Polygen: An autoregressive generative model of 3d meshes. *ICML*, 2020. [2](#)
- [36] 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](#), [2](#)
- [37] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. [2](#), [3](#), [18](#)
- [38] 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 (CVPR)*, pages 13503–13513, June 2022. [2](#)
- [39] 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. [4](#), [14](#)
- [40] Dario Pavllo, Jonas Kohler, Thomas Hofmann, and Aurelien Lucchi. Learning generative models of textured 3d meshes from real-world images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13879–13889, October 2021. [2](#)
- [41] Dario Pavllo, Graham Spinks, Thomas Hofmann, Marie-Francine Moens, and Aurelien Lucchi. Convolutional generation of textured 3d meshes. *Advances in Neural Information Processing Systems*, 33:870–882, 2020. [2](#)
- [42] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv*, 2022. [1](#), [3](#), [8](#)
- [43] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. [2](#)
- [44] Tingting Qiao, Jing Zhang, Duanqing Xu, and Dacheng Tao. Mirrorgan: Learning text-to-image generation by redescription. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. [2](#)
- [45] 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](#)
- [46] 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](#), [2](#), [12](#)
- [47] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021. [1](#), [2](#)
- [48] Scott Reed, Zeynep Akata, Xinchun Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *International conference on machine learning*, pages 1060–1069. PMLR, 2016. [2](#)
- [49] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015. [3](#)
- [50] 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. [1](#), [2](#), [3](#), [5](#)

- [51] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. *arXiv preprint arXiv:2208.12242*, 2022. [2](#), [5](#), [6](#)
- [52] 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](#), [2](#), [3](#)
- [53] Aditya Sanghi, Hang Chu, Joseph G. Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshah. Clip-forge: Towards zero-shot text-to-shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18603–18613, June 2022. [1](#), [2](#), [3](#), [4](#), [5](#), [7](#), [8](#), [14](#)
- [54] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022. [2](#), [5](#)
- [55] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *Advances in Neural Information Processing Systems*, 33:20154–20166, 2020. [2](#)
- [56] Dong Wook Shu, Sung Woo Park, and Junseok Kwon. 3d point cloud generative adversarial network based on tree structured graph convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. [14](#)
- [57] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. [2](#), [5](#)
- [58] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5459–5469, June 2022. [3](#), [4](#), [13](#), [14](#)
- [59] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. [2](#)
- [60] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. *arXiv preprint arXiv:2212.00774*, 2022. [8](#)
- [61] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. [15](#)
- [62] Zihao Wang, Wei Liu, Qian He, Xinglong Wu, and Zili Yi. Clip-gen: Language-free training of a text-to-image generator with clip. *arXiv preprint arXiv:2203.00386*, 2022. [1](#)
- [63] Rundi Wu and Changxi Zheng. Learning to generate 3d shapes from a single example. *ACM Transactions on Graphics (TOG)*, 41(6), 2022. [2](#)
- [64] Rundi Wu, Yixin Zhuang, Kai Xu, Hao Zhang, and Baoquan Chen. Pq-net: A generative part seq2seq network for 3d shapes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. [2](#)
- [65] Peng Xiang, Xin Wen, Yu-Shen Liu, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Zhizhong Han. Snowflake point deconvolution for point cloud completion and generation with skip-transformer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. [2](#)
- [66] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. [2](#)
- [67] 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 (ICCV)*, October 2019. [2](#)
- [68] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems*, 34:4805–4815, 2021. [4](#), [14](#)
- [69] 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. [2](#)
- [70] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. [2](#)
- [71] Xin-Yang Zheng, Yang Liu, Peng-Shuai Wang, and Xin Tong. Sdf-stylegan: Implicit sdf-based stylegan for 3d shape generation. In *Comput. Graph. Forum (SGP)*, 2022. [2](#), [5](#), [6](#), [12](#), [13](#)
- [72] 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 (ICCV)*, pages 5826–5835, October 2021. [2](#)

## A. Details of 3D Generator $G_S$

We adopt the architecture of SDF-StyleGAN [71] as our 3D generator. As Fig. 7 shows, it maps a random noise  $z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  to a latent shape embedding  $e_S \in \mathcal{W}$  and synthesizes a 3D feature volume  $F_V$ , which is an implicit representation of the generated shape. We can query the SDF value at arbitrary position  $\mathbf{x}$  by feeding the interpolated feature from  $F_V$  at  $\mathbf{x}$  into a jointly trained MLP network. During training, a global discriminator and a local discriminator are used simultaneously to supervise the generated SDF grids at the coarse and fine level respectively. Different from the original SDF-StyleGAN that trains one network for *one shape category*, we train *one* 3D shape generator  $G_S$  on *13 categories* of the ShapeNet [6] dataset to enlarge the shape generation capability.

## B. Details of Shape Embedding Mapping Network $G_M$

The shape embedding mapping network  $G_M$  is a diffusion-model-based generative network that can generate shape embeddings  $e_S$  from the CLIP image embeddings  $e_I$  of shape renderings. The network architecture and training strategy of  $G_M$  are based on an open-source DALL-E-2 [46] implementation<sup>§</sup>. Specifically,  $G_M$  is equivalent to the *diffusion prior network* in DALL-E-2 which generates CLIP image embeddings from CLIP text embeddings. Here we replace the input with CLIP image embeddings of shape renderings and the output with shape embeddings. We use the *DiffusionPrior* class in the codebase to implement  $G_M$  and the *train\_diffusion\_prior.py* script to train  $G_M$ . The model and training hyperparameters are listed in Tab. 3.

## C. Details of Fine-tuning Stable Diffusion

In our framework, we connect the text and image modalities by fine-tuning the Stable Diffusion into a stylized generator with a set of shape renderings  $\{I_S^j\}_{j=1}^{N_S}$  and give it the ability to synthesize images in the “rendering” style. In experiments, we find it crucial to utilize a large set of shape renderings for fine-tuning and to augment the backgrounds of the shape renderings with random colors.

We tried fine-tuning Stable Diffusion using shape renderings with three different types of backgrounds: 1) solid white background, 2) solid green background and 3) random-color background. We visualize the shape renderings used for fine-tuning and the images synthesized by the fine-tuned Stable Diffusion in Fig. 8. As Fig. 8a and Fig. 8b show, although shape renderings with solid-white or solid-green backgrounds can make the fine-tuned Stable Diffusion capture the “rendering” style of the object successfully, the backgrounds in the synthesized images are

<sup>§</sup><https://github.com/lucidrains/DALLE2-pytorch>

Model Parameter	Value	Training Parameter	Value
timesteps	100	iterations	500,000
beta_schedule	cosine	max_grad_norm	0.5
predict_x_start	True	batch_size	1024
cond_drop_prob	0	learning_rate	$1.1 \times 10^{-4}$
dim	512	weight_decay	$6.02 \times 10^{-2}$
depth	6	ema_beta	0.9999
dim_head	64	ema_update_every	10
heads	8	Adam $\beta_1, \beta_2$	0.9, 0.999

Table 3. Model details and training hyper-parameters of the shape embedding mapping network  $G_M$ .

Background	FID ↓
Solid white	60.61
Solid green	71.04
Random-color	<b>33.71</b>

Table 4. The Fréchet Inception Distance (FID) between the shape renderings used for fine-tuning and the images synthesized by the fine-tuned Stable Diffusion.

out of control, *i.e.*, the fine-tuned Stable Diffusion fails to synthesize images with solid-color backgrounds. This will increase the difficulty of separating the foreground objects from the backgrounds and affect the stability of the subsequent image-to-shape generation since the shape embedding mapping network  $G_M$  is trained on shape renderings with solid-color backgrounds. In comparison, augmenting the backgrounds of the shape renderings with random colors leads to a stable stylized generator that can synthesize solid-color-background images consistently, as Fig. 8c shows.

During fine-tuning, we indeed expect the Stable Diffusion model to capture two types of styles: 1) the “rendering” style of the foreground object and 2) the “solid-color” style of the background. Similar to the observation that the foreground “rendering” style requires a large set of rendered images to learn, we consider that a single-color background is too few to be recognized as a “solid-color background style” by the Stable Diffusion model, while showing a lot of different solid-color examples to the model can make it notice the solid-color background style and capture it during fine-tuning.

To better demonstrate the importance of the random-color background augmentation, we also evaluate the Fréchet Inception Distance (FID) between the shape renderings used for fine-tuning and the images synthesized by the fine-tuned Stable Diffusion in Tab. 4. For each type of background, we render 1000 images with that background for each ShapeNet [6] category, forming a shape rendering dataset containing 13000 images in total (denoting as  $D_S$ ). Then we leverage  $D_S$  to fine-tune the Stable Diffusion model for 5000 steps, and utilize the fine-tuned Stable Diffusion to synthesize 100 images for each shape category using the text prompt “a CLS in the style of\*”, lead-

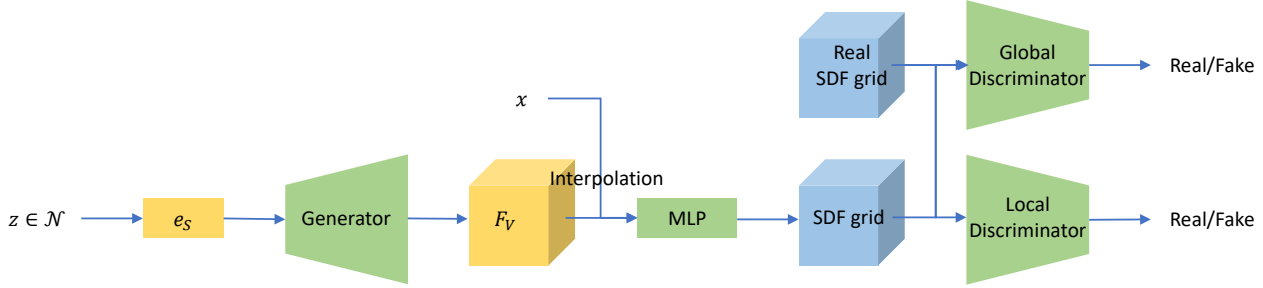


Figure 7. The network architecture of the 3D generator based on SDF-StyleGAN [71].

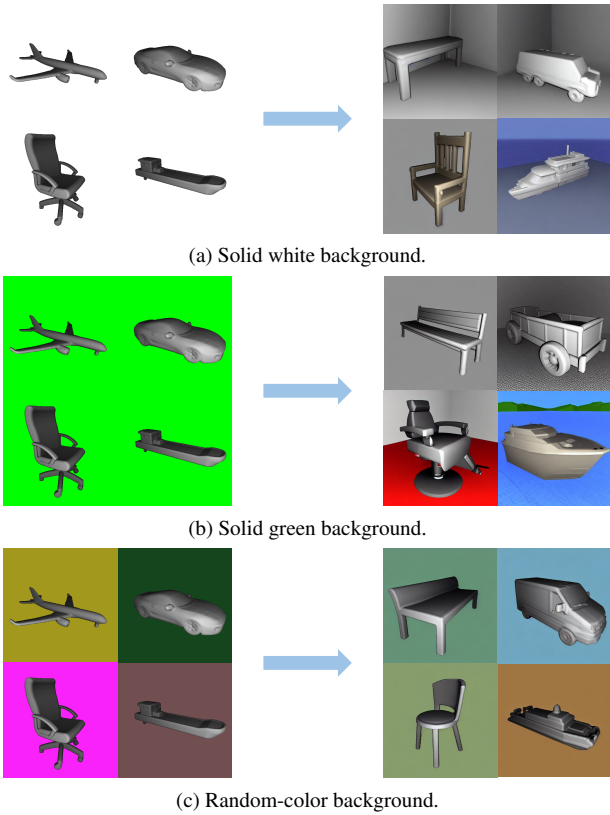


Figure 8. The results of fine-tuning Stable Diffusion using shape renderings with different backgrounds. For each type of background, we visualize the shape renderings used to fine-tune Stable Diffusion on the left and the images synthesized by the fine-tuned Stable Diffusion on the right.

ing to a set of 1300 generated images (denoting as  $D_{gen}$ ). Finally, we compute the FID between  $D_S$  and  $D_{gen}$ . As Tab. 4 shows, augmenting the backgrounds of shape renderings with random colors significantly boosts the FID, which demonstrates its effectiveness.

## D. Details of 3D Optimization with 3D Shape Prior

### D.1. DVGO-based Volume Rendering

In the optimization stage, we adopt DVGO [58] as our 3D scene representation which represents NeRF [34] with a density voxel grid  $\mathbf{V}^{density} \in \mathbb{R}^{N_x \times N_y \times N_z}$  and a shallow color MLP network  $f_{rgb}$  for efficient optimization. Given a 3D position  $\mathbf{x}$ , we query its density  $\sigma$  and color  $c$  by:

$$\tilde{\sigma} = \text{interp}(\mathbf{x}, \mathbf{V}^{density}), \quad (5a)$$

$$\sigma = \text{softplus}(\tilde{\sigma}) = \log(1 + \exp(\tilde{\sigma} + b)), \quad (5b)$$

$$c = f_{rgb}(\gamma(\mathbf{x})), \quad (5c)$$

where  $\gamma(\cdot)$  denotes a positional encoding function.  $\text{interp}(\cdot)$  denotes the trilinear interpolation. The shifted softplus function  $\text{softplus}(\cdot)$  is applied to transform the raw density value  $\tilde{\sigma}$  into activated density value  $\sigma$  (i.e., a mapping of  $\mathbb{R} \rightarrow \mathbb{R}_{\geq 0}$ ), the shift  $b$  is a hyperparameter. To be noted, the density grid  $\mathbf{V}^{density}$  stores the raw density values instead of the activated ones. DVGO [58] calls the scheme of interpolating on the raw density values first and then performing softplus activation as "post-activation" and demonstrates its advantages on producing sharper shape boundaries over other choices.

To render the color of a pixel  $\hat{C}(r)$ , we cast the ray  $r$  from the camera center through the pixel, and sample  $K$  points between the pre-defined near and far planes. We then query the densities and colors of the  $K$  ordered sampled points  $\{(\sigma_i, c_i)\}_{i=1}^K$  using Eq. (5). Finally, we accumulate the  $K$  queried results into a single color with the volume rendering process:

$$\hat{C}(r) = \left( \sum_{i=1}^K T_i \alpha_i c_i \right) + T_{K+1} c_{bg}, \quad (6a)$$

$$\alpha_i = \text{alpha}(\sigma_i, \delta_i) = 1 - \exp(-\sigma_i \delta_i), \quad (6b)$$

$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j), \quad (6c)$$

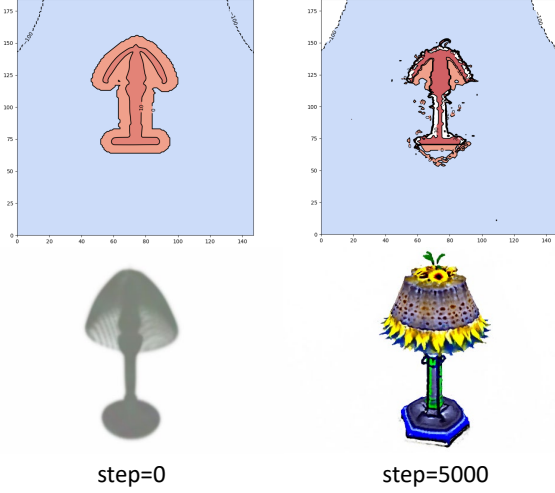


Figure 9. Illustration on 3D optimization with 3D shape prior. The text prompt is "A lamp imitating sunflower". We visualize the contours of the density grid (up) and the volume-rendered images (bottom) at the 0<sup>th</sup> and 5000<sup>th</sup> optimization steps to show how the density grid is initialized and optimized.

where  $\alpha_i$  denotes the opacity representing the probability of termination at point  $i$ ,  $T_i$  denotes the accumulated transmittance from the near plane to point  $i$ ,  $\delta_i$  denotes the distance to the adjacent sampled point, and  $c_{bg}$  demotes a pre-defined background color.

Following DVGO, all values in  $\mathbf{V}_{density}$  are initialized as 0 and the bias term in Eq. (5b) is set to

$$b = \log \left( (1 - \alpha_{init})^{-\frac{1}{s}} - 1 \right), \quad (7)$$

where  $\alpha_{init}$  is a hyperparameter and is set to  $10^{-6}$  in practice. With such an initialization, the accumulated transmittance  $T_i$  is decayed by  $1 - \alpha_{init} \approx 1$  for a ray that traces forward a distance of a voxel size  $s$ , making the scene "transparent" at the beginning of optimization.

## D.2. Shape Prior Initialization and Optimization

A big difference between our text-guided 3D synthesis framework and previous methods [20, 23, 25] is that we use an explicit 3D shape prior to initialize the CLIP-guided optimization process, instead of optimizing from a randomly-initialized 3D representation. Given a 3D shape prior  $S$  represented by an SDF grid  $\tilde{\mathbf{V}}_{sdf} \in \mathbb{R}^{N_x \times N_y \times N_z}$ , we use it to initialize the density voxel grid  $\mathbf{V}_{density}$  with the following equations [39, 58, 68]:

$$\Sigma = \frac{1}{\beta} \text{sigmoid} \left( -\frac{\tilde{\mathbf{V}}_{sdf}}{\beta} \right), \quad (8a)$$

$$\mathbf{V}_{density} = \max(0, \text{softplus}^{-1}(\Sigma)), \quad (8b)$$

Method	FID ↓	FPD ↓	MMD ↑
CLIP-Forge [53]	112.38	6.896	0.670
Ours	<b>40.83</b>	<b>1.301</b>	<b>0.725</b>

Table 5. Additional quantitative results compared with CLIP-Forge on text-guided shape generation.

where  $\text{sigmoid}(x) = 1/(1 + e^{-x})$  and  $\text{softplus}^{-1}(x) = \log(e^x - 1)$ . Eq. (8a) converts SDF values to activated density values (equivalent to the  $\sigma$  in Eq. (5b)), where  $\beta > 0$  controls the sharpness of the shape boundary, and smaller  $\beta$  leads to a sharper shape boundary. We set  $\beta = 0.05$  in our experiments. Eq. (8b) further transforms the activated density values into raw density values (equivalent to the  $\tilde{\sigma}$  in Eq. (5a)).

With such an initialization, the density values on the shape surface will be close to  $\frac{1}{2\beta} (\log(\exp(\frac{1}{\beta} \cdot \text{sigmoid}(0)) - 1) \approx \frac{1}{2\beta})$ . The area inside the shape surface will have larger density values ( $> \frac{1}{2\beta}$ ), and the density values outside the shape will decrease with the distance from the shape surface. We set the minimum density value outside the shape to 0 so the area far from the shape surface has the same initialization as the original DVGO.

As Fig. 9 shows, at the beginning of the 3D optimization process (step=0), the 3D shape prior is visible due to the larger density values around the shape surface. As a result, the area around the shape surface will dominate the volume rendering, and the density/color values in this area will be updated faster than the area far from the surface. Based on the initialization, Then subsequent CLIP-guided optimization process further provides more flexibility and is able to synthesize more diverse structures and textures.

## E. Additional Results on Text-to-Shape Generation

We show additional qualitative text-guided 3D shape generation results in Fig. 10. Compared to CLIP-Forge [53], our method produces more plausible 3D shapes thanks to the high-quality 3D generator, while the shapes generated by [53] suffer from rough surfaces and discontinuities.

Besides, we also provide more quantitative comparisons with CLIP-Forge on text-to-shape generation. We generate 3 shapes for each text prompt in the text prompt set provided by CLIP-Forge and measure three metrics: 1) Fréchet Inception Distance (FID) [18] between 5 rendered images for each shape with different camera poses and a set of images rendered from the ground truth shapes in the ShapeNet dataset with the same camera poses. 2) Fréchet Point Distance (FPD) [56], for each generated shape and each ground truth shape in the ShapeNet test set, we extract the mesh at  $64^3$  resolution and sample 2048 points from the mesh sur-

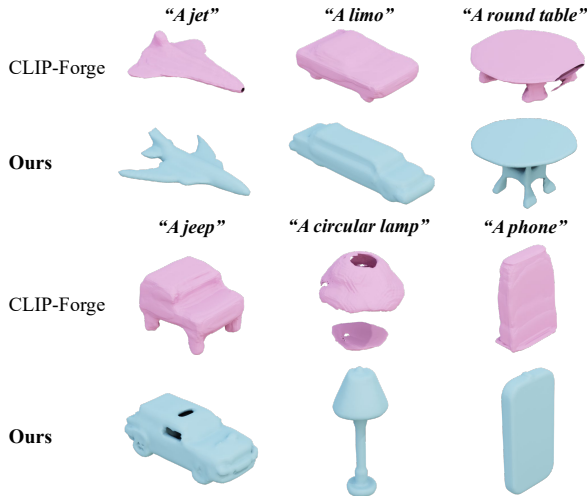


Figure 10. Additional text-guided shape generation results. All meshes are extracted at  $64^3$  resolution.

face, then pass the points to a DGCNN [61] backbone network pre-trained on the point cloud classification task and use the feature of the last layer to compute this metric. 3) Maximum Measure Distance (MMD), for each generated shape represented by a  $32^3$  occupancy grid, we match a shape in the ShapeNet test set based on the highest IOU, and then average the IOU across all the text queries. As Tab. 5 shows, our text-to-shape generation method outperforms CLIP-Forge on all three metrics.

## F. Additional Results on Text-to-3D Synthesis

In this section, we show additional qualitative comparison results on text-to-3D synthesis with baseline methods in Fig. 11 and more diversified generation results of our method in Fig. 12. It can be seen that our method can synthesize plausible 3D structures with the help of 3D shape priors. To better visualize the 3D structures generated by different methods, we also show video examples in the attached MP4 file.



Figure 11. Additional qualitative comparisons on text-guided 3D synthesis. For each of our results (the last row), we also visualize the 3D shape prior used to initialize the CLIP-guided optimization process below it.



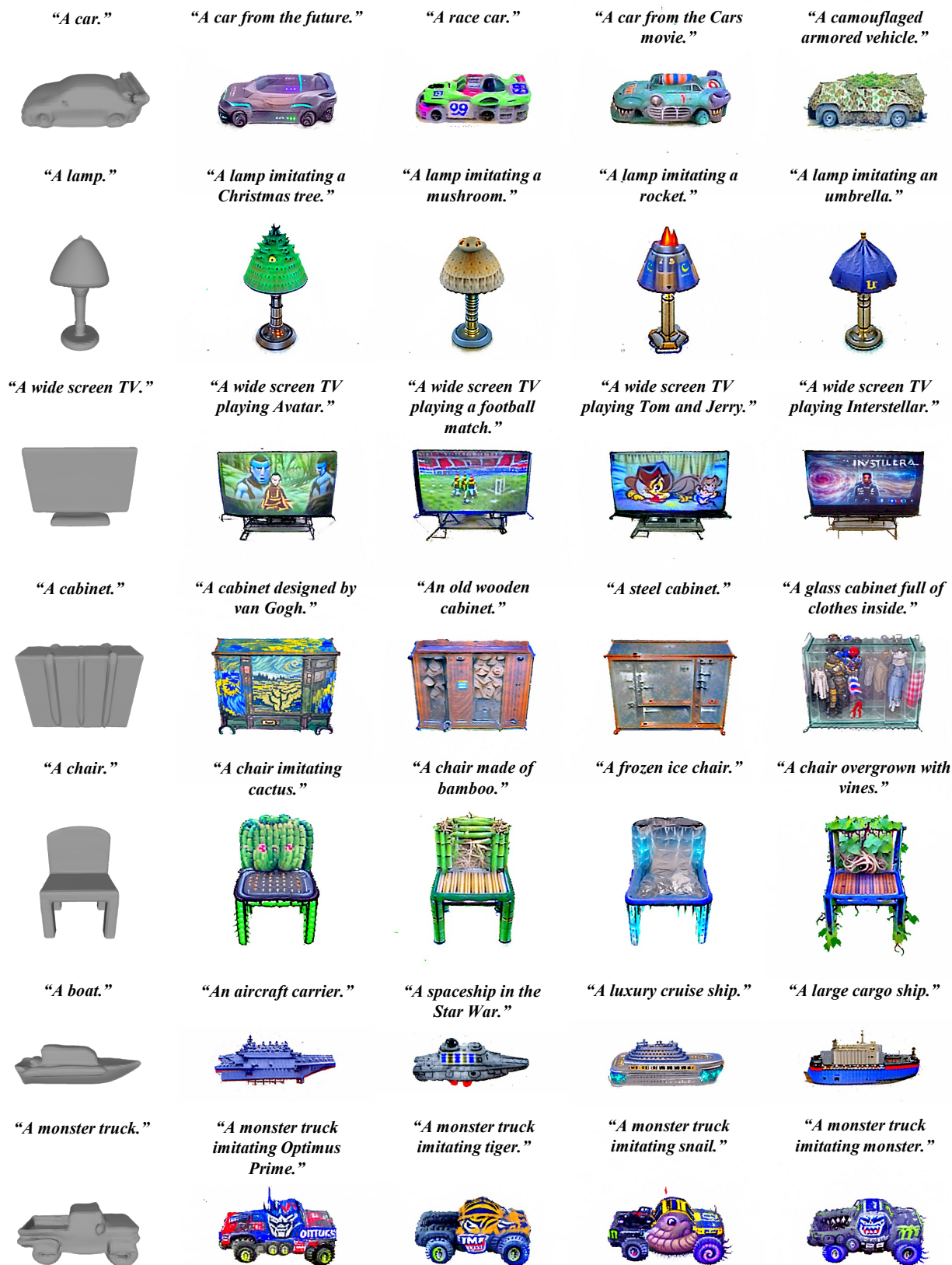


Figure 12. Additional text-to-3D synthesis results. We visualize the 3D shape prior used for optimization in the first column.

## G. Integration with SVR Models

### G.1. Text-to-Shape Generation using SVR models

Single-view reconstruction (SVR) models can reconstruct a 3D shape from a single input image. We then ask, can we use an SVR model directly as the image-to-shape module in our framework? To answer this question, we conduct the same fine-tuning process to fine-tune a Stable Diffusion model with the ShapeNet renderings provided by Choy *et al.* [9] which are commonly used by many SVR methods. We find that although the shape renderings in Choy *et al.* [9] have more complex textures, the fine-tuned model can still capture the style successfully and synthesize novel images imitating the style. With such a fine-tuned Stable Diffusion, we can solve the text-to-shape generation in a precise way: synthesize an image using the fine-tuned Stable Diffusion with text prompt in the format of "a *CLS* in the style of \*", and then directly feed the synthesized image into the SVR model. We show some text-guided shape generation results using two SVR methods, *i.e.*, occupancy networks [31] and DVR [37], in Fig. 13 and Fig. 14, respectively. Both methods are trained with the shape renderings provided by Choy *et al.* [9]. The occupancy networks only predict shape, while DVR can predict both shape and color. As Fig. 13 and Fig. 14 show, we achieve text-to-shape generation successfully with the synthesized images, which proves the strong generation ability of Stable Diffusion and the effectiveness of our fine-tuning pipeline.

A recent work named ISS [28] also utilizes an SVR model to perform text-to-shape generation. However, the pipeline of ISS is much more complicated. It trains a mapper network to map CLIP features to the latent space of the SVR model, which requires a two-stage fine-tuning to align the text and shape feature spaces. At inference time, ISS needs to fine-tune the mapper network for each text prompt, which is redundant in our pipeline. With the help of the fine-tuned Stable Diffusion, we can directly generate an image from the text prompt and feed the image into the SVR model to synthesize a 3D shape. Besides, thanks to the strong generation ability of Stable Diffusion, we can enjoy a much larger generation diversity and synthesize as many 3D shapes as we want for each text prompt.

### G.2. Text-to-3D Synthesis using SVR models

Despite the success in text-guided shape generation with SVR models, we find that current SVR models are very sensitive to the input images. Although we can successfully capture the style of the shape renderings using the fine-tuned Stable Diffusion, some minor flaws in the synthesized images such as offsets of the objects from the image center and unrealistic artifacts (*e.g.*, a chair lacks a leg) are inevitable. These minor flaws may lead to failed shape reconstructions, whose quality affects 3D shape priors. This



Figure 13. Text-guided shape generation using fine-tuned Stable Diffusion and Occupancy Networks [31]. For each text prompt, we visualize the image synthesized by the fine-tuned Stable Diffusion on the left and the reconstructed shape on the right.

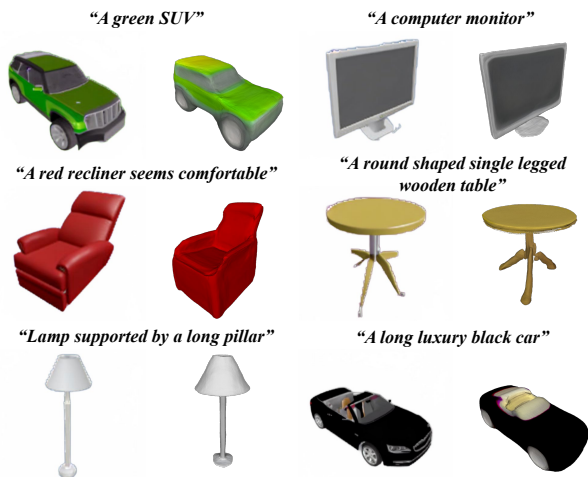


Figure 14. Text-guided shape generation using fine-tuned Stable Diffusion and DVR [37]. For each text prompt, we visualize the image synthesized by the fine-tuned Stable Diffusion on the left and the reconstructed shape on the right.

sensitiveness makes the 3D prior generation in the first stage of our framework unstable. Therefore, we choose to use a 3D generator associated with a shape embedding mapping network to generate 3D shapes in the latent shape embedding space, instead of directly using an SVR model in our framework.

We visualize six text-to-3D synthesis results using 3D shape priors produced by the occupancy networks [31] in Fig. 15. The successful results in the first two rows show the probability of integrating as SVR model into our framework. In the last row, we show two failure cases in which the SVR model fails to reconstruct plausible 3D shape pri-

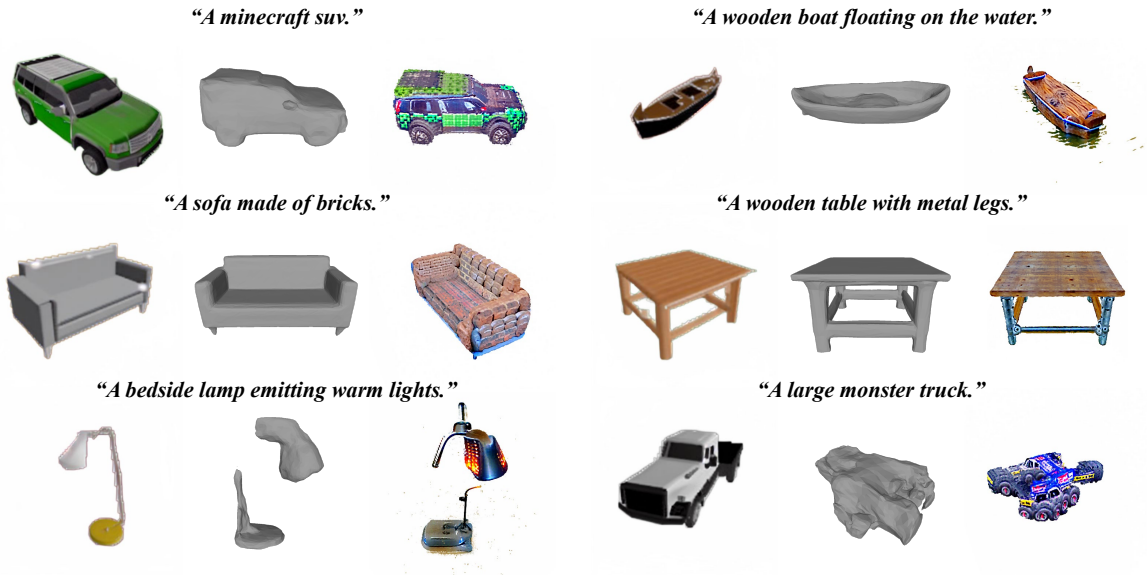


Figure 15. Text-to-3D synthesis results using occupancy networks. For each text prompt, we visualize the shape rendering image synthesized by the fine-tuned Stable Diffusion on the left, the shape reconstructed by the SVR model in the middle, and the optimization result on the right. The last row shows two failure cases.

ors to illustrate the drawbacks of using SVR models. We can observe that the discontinuity in the “bedside lamp” shape leads to discontinuity in the final optimization result, while the failed truck shape results in total chaos.