BoostDream: Efficient Refining for High-Quality Text-to-3D Generation from Multi-View Diffusion

Yonghao Yu[1*], **Shunan Zhu**[1], **Huai Qin**[1] and **Haorui Li**[2]
[1]Waseda University
[2] Southeast University
yuyonghao@suou.waseda.jp, {shunan-zhu, mizuki_qin}@ruri.waseda.jp, lihaorui.lhr@alibaba-inc.com

Abstract

Witnessing the evolution of text-to-image diffusion models, significant strides have been made in text-to-3D generation. Currently, two primary paradigms dominate the field of text-to-3D: the feed-forward generation solutions, capable of swiftly producing 3D assets but often yielding coarse results, and the Score Distillation Sampling (SDS) based solutions, known for generating high-fidelity 3D assets albeit at a slower pace. The synergistic integration of these methods holds substantial promise for advancing 3D generation techniques. In this paper, we present BoostDream, a highly efficient plug-and-play 3D refining method designed to transform coarse 3D assets into high-quality. The BoostDream framework comprises three distinct processes: (1) We introduce 3D model distillation that fits differentiable representations from the 3D assets obtained through feed-forward generation. (2) A novel multi-view SDS loss is designed, which utilizes a multi-view aware 2D diffusion model to refine the 3D assets. (3) We propose to use prompt and multi-view consistent normal maps as guidance in refinement. Our extensive experiment is conducted on different differentiable 3D representations, revealing that BoostDream excels in generating high-quality 3D assets rapidly, overcoming the Janus problem compared to conventional SDS-based methods. This breakthrough signifies a substantial advancement in both the efficiency and quality of 3D generation processes.

1 Introduction

The significance of 3D assets has grown remarkably due to the expansion of virtual reality and the gaming industry. Creating these assets, however, involves considerable labor costs. Recently, the popularity of methods based on differentiable rendering [Mildenhall *et al.*, 2021; Shen *et al.*, 2021; Wang *et al.*, 2021; Kerbl *et al.*, 2023] and the explosion of text-to-image models [Rombach *et al.*, 2022; Zhang *et al.*, 2023a] provides a new line of methods for generating 3D assets.

Plenty of research [Nichol *et al.*, 2022; Jun and Nichol, 2023a; Poole *et al.*, 2022; Wang *et al.*, 2023b; Lin *et al.*, 2023; Chen *et al.*, 2023b] in the text-to-3D domain aim to lower 3D asset creation costs, demonstrating the efficacy of leveraging 2D generation models for 3D tasks. Currently, there are two primary strategies in the field of text-to-3D generation. The first, exemplified by Point-E and Shap-E [Nichol *et al.*, 2022; Jun and Nichol, 2023a], employs a feed-forward approach, using 3D datasets or latent representation mapped by the 3D datasets to train diffusion models. This enables quick, text-prompt-based 3D asset generation. However, these methods grapple with the limited variety and scope of 3D datasets.

In response to the constraints posed by limited 3D datasets, the SDS-based optimization method was developed. Models like DreamFusion and Magic3D [Poole *et al.*, 2022; Lin *et al.*, 2023] exemplify this approach. They employ an SDS loss function [Poole *et al.*, 2022], enabling text-to-image diffusion models to train 3D assets predominantly represented by methods based on differentiable rendering, such as NeRF [Mildenhall *et al.*, 2021]. This approach allows for the generation of high-quality 3D assets, leveraging the strong text comprehension and 2D generation abilities of diffusion models. However, this method is hindered by slow generation speeds, as each text prompt requires training a 3D representation from scratch and suffering from the Janus (multi-head) problem. How to efficiently generate high-quality 3D assets remains an urgent problem to be solved.

With this in mind, we propose a generation method suitable for various 3D representations, utilizing the rapid generation of the feed-forward method and the high-quality generation characteristics of the SDS-based method. Our method does not just simply combine the feed-forward approach with the SDS-based method. Specifically, we have developed an innovative three-stage method named BoostDream. In the first stage, we designed a rapid initialization technique that transforms explicitly represented 3D assets generated by the feed-forward method into 3D representations based on differentiable rendering. In the second stage, we introduced a multi-view rendering system and a multi-view SDS under the control of the original 3D input condition. In the third stage, we solely rely on self-supervision to achieve refined generation results. Extensive experiments are conducted, including

*Corresponding author

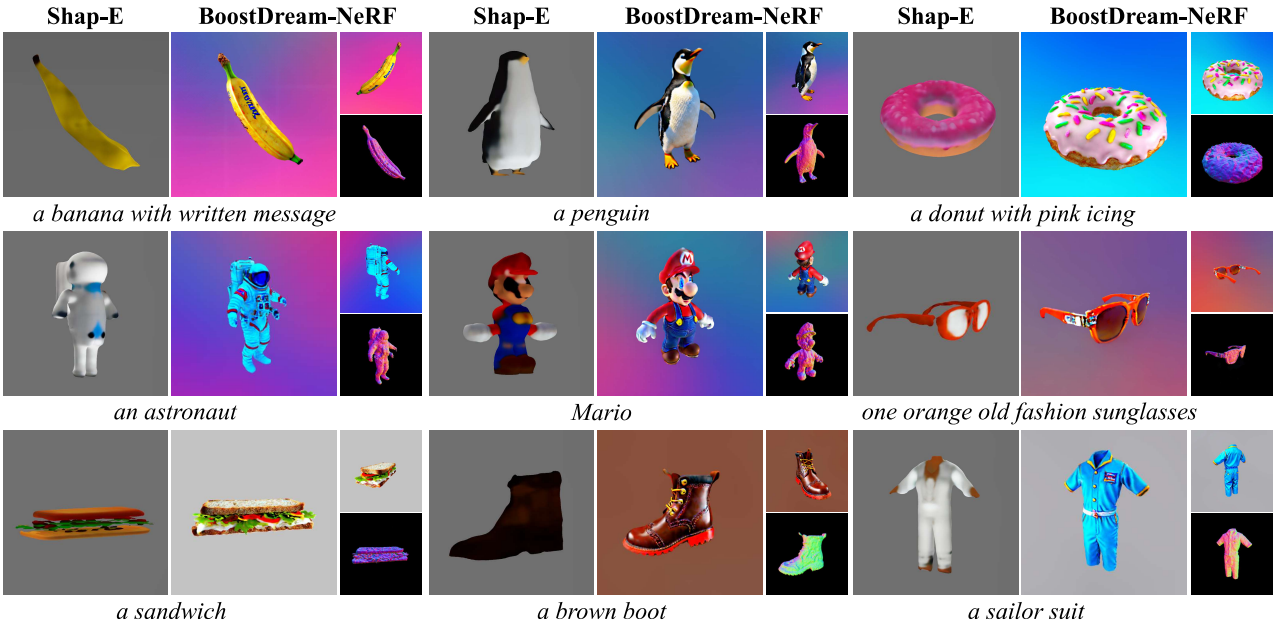


Figure 1: **Comparison of 3D Generation Results of baseline and *BoostDream*.** Provided with a coarse 3D asset and text prompt pair, BoostDream can refine it into a high-quality 3D asset efficiently. In each set of images, the image on the left is the coarse 3D asset generated by Shap-E [Jun and Nichol, 2023a], and the three images on the right are our refined 3D asset.

refinement and comparison experiments. As shown in Figure 1, our method can efficiently refine the coarse 3D for high-quality results. In summary, the contributions of this paper include the following:

- We propose a novel method that integrates the advances in feed-forward and SDS-based methods, enabling efficient and high-quality refinement of 3D assets.
- We innovatively propose the multi-view SDS with the multi-view render system to refine 3D assets under multi-view conditions and address the Janus problem.
- Our BoostDream can generate high-quality 3D assets based on a variety of 3D differentiable representations, showcasing strong generalizability.

2 Related Work

In this section, we discuss the related work on text-to-3D generation using diffusion models. One mainstream approach is the feed-forward method using 3D datasets for training, and another is SDS-based optimization. In addition, the multi-view perceptual diffusion method provides ideas for solving the Janus problem and has proved effective.

2.1 Feed-Forward Generation Method

Feed-forward generation method tries to directly utilize the diffusion models in text-to-3D tasks, which trained diffusion models with 3D assets as datasets [Liu *et al.*, 2023c; Liu *et al.*, 2023a; Sanghi *et al.*, 2023; Nichol *et al.*, 2022; Zhou *et al.*, 2021] or perform diffusion with a latent representation of 3D assets [Gupta *et al.*, 2023; Ntavelis *et al.*, 2023; Chen *et al.*, 2023a; Zeng *et al.*, 2022]. Shap-E [Jun and Nichol, 2023a] is a typical example of this type of approach. It trains a 3D encoder to transform 3D assets to a set of parameters of an implicit function and then trains a diffusion model on these parameters. Compared to the methods using frozen 2D diffusion models, these approaches can generate 3D assets at a very fast speed due to their feed-forward nature. However, these methods were trained with the 3D datasets, while the 3D dataset available today is relatively small and low-quality. Consider the dataset used in 2D image generation tasks, Laion5B [Schuhmann *et al.*, 2022] contains more than 5 billion image-text pairs while the largest 3D dataset available, Objaverse-XL [Deitke *et al.*, 2023] can only carry 10 million 3D assets with worse quality captions. This causes the 3D generation results to be less sharp, have lower fidelity, and be unable to produce assets with complex semantics.

2.2 SDS-Based Optimization Generation Method

Inspired by the great success of using diffusion models in text-to-image tasks, one mainstream of works focuses on using 2D diffusion models along with differentiable 3D representing techniques like NeRF [Mildenhall *et al.*, 2021] to facilitate 3D generation. Dreamfields [Jain *et al.*, 2022] used pre-trained CLIP models to optimize NeRF, hoping to get a visually real result. But it takes much time and computing resources to generate. DreamFusion [Poole *et al.*, 2022] proposes SDS loss to optimize NeRF representation. SJC [Wang *et al.*, 2023a] is a concurrent effort of Dreamfusion, with the difference that SJC applies the chain rule to the 2D score. Research on providing supervision of novel views via SDS loss is becoming a trend [Lin *et al.*, 2023; Chen *et al.*, 2023b; Metzer *et al.*, 2023; Liu *et al.*, 2023b; Qian *et al.*, 2023; Melas-Kyriazi *et al.*, 2023; Lorraine *et al.*, 2023]. Among them, ProlificDreamer [Wang *et al.*, 2023b] models the 3D parameters as variable, proposes variable fraction distillation

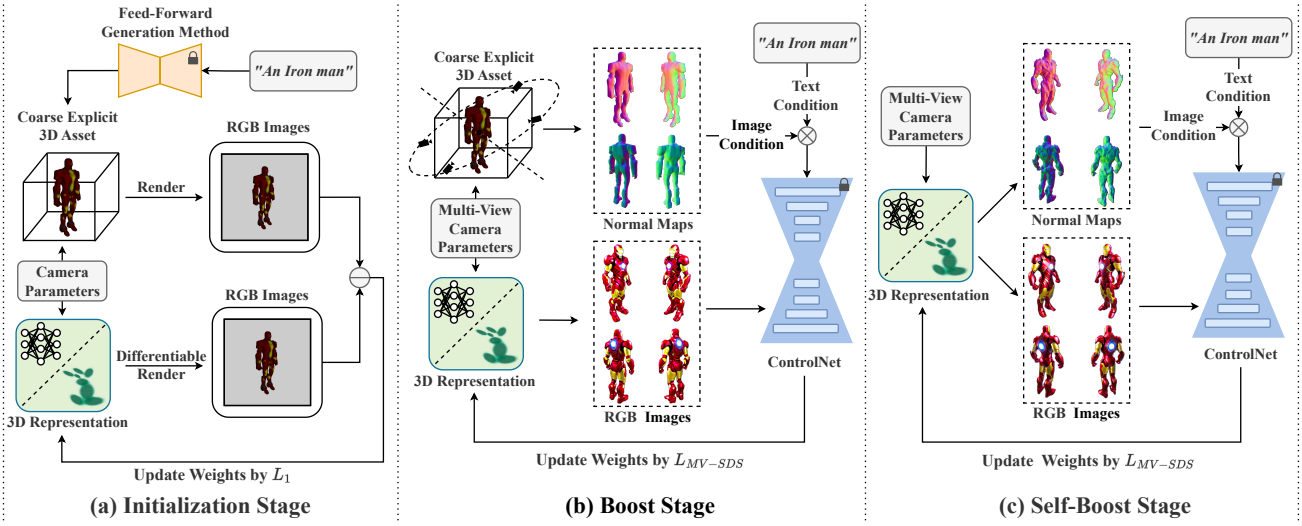


Figure 2: **Overview of the proposed *BoostDream***. BoostDream is a three-stage framework for refining a coarse 3D asset into a high-quality 3D asset. In the initialization stage, we use the feed-forward generation method to get a coarse 3D asset and fit it into differentiable 3D representations to make it trainable. The boost stage is guided by the multi-view normal maps of the coarse 3D asset to ensure stability from the beginning of the refining stage, and the self-boost stage is guided by its own multi-view normal maps to generate 3D assets with more detail and higher quality.

(VSD) to address the shortcomings in SDS. These works indicate the capability to produce high-quality outputs enriched with detailed semantic information derived from 2D diffusion models. However, these SDS-based methods are impeded by prolonged optimization periods and the occurrence of Janus problems.

2.3 Multi-View in 3D Generation

The multi-view aware diffusion method in image generation has made great progress recently. MVDiffusion [Tang *et al.*, 2023] is capable of generating consistent multi-view images from text prompts given pixel-to-pixel correspondences with global awareness. The success of such a method also draws attention to 3D generation tasks. EfficientDreamer [Zhao *et al.*, 2023] used a 2D diffusion model that can generate an image consisting of four orthogonal-view sub-images and use it in the 3D generation. It has been proven to be efficient in improving the quality of 3D content and alleviating the Janus problem. MVDream [Shi *et al.*, 2023] further discusses the situation of building a multi-view diffusion model from both 2D and 3D data. It can benefit from the generalizability of the 2D diffusion model while keeping the consistency of 3D rendering, solving the Janus problem and content drift problem across different views. However, after training by the 3D datasets, the realistic texture information that the 2D diffusion model contained can be degraded. In our work, a multi-view rendering method is designed and a multi-view SDS loss function is formulated to guide the differentiable 3D representation, mitigating the Janus problem while obtaining high-quality generated results.

3 BoostDream

In this section, we propose our BoostDream model. We initially present a brief overview of the background of text-to-3D generation. Subsequently, we introduce our multi-view refining approach, detailing its design and implementation within the context of enhanced 3D asset generation by quick initialization based on differentiable rendering, multi-view render system, and multi-view SDS. Figure 2 illustrates the overview of our method.

3.1 Background

For text-to-image tasks, [Ho *et al.*, 2020] has proposed a simplified training goal of Denoising Diffusion Probabilistic Models (DDPM). It is expressed as:

$$L_{\text{DDPM}} = \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0+\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon},t\right)\right\|^2\right] \tag{1}$$

Here the x_t is latent at time step t, ϵ is the actual noise, ϵ_θ represents the predicted noise, $\alpha_t = 1-\beta_t$ where β_t is the noise added at time step t in a diffusion model.

In text-to-3D tasks using text-prompt as the generation condition, DreamFusion [Poole *et al.*, 2022; Wang *et al.*, 2023b] has proposed the SDS loss. For a differentiable 3D rendering represented as $g(\theta, c)$, where θ are its parameters and c are the camera parameters. Let $q_t^\theta(\mathbf{x}_t \mid c)$ be the distribution from diffusing $\mathbf{x}_0 = g(\theta, c)$ to moment t. The optimization goal of SDS is by optimizing θ, make the marginal distribution of the diffusion process of $q_t^\theta(\mathbf{x}_t \mid c)$, approach the marginal distribution of a pre-trained text-to-image model $p(\mathbf{x}_t \mid y)$ where y is text-prompt. The training goal of the SDS loss method can be represented as:

$$\min_\theta \mathcal{L}_{\text{SDS}} = \mathbb{E}_{t,c}\left[(\sigma_t/\alpha_t)\, w(t) D_{KL}\left(q_t^\theta(\mathbf{x}_t \mid c) \,\|\, p(\mathbf{x}_t \mid y)\right)\right] \tag{2}$$

The SDS method estimates the gradient of the optimization object $\mathcal{L}_{\text{SDS}}$ as follows, allowing the text-to-3D task to be

carried out quickly and without losing too much precision:

$$\nabla_\theta \mathcal{L}_{\text{SDS}} = \mathbb{E}_{t,\epsilon,c}\left[w(t)\left(\epsilon_\theta\left(\mathbf{x}_t, t, y\right) - \epsilon\right)\frac{\partial g(\theta, c)}{\partial \theta}\right] \quad (3)$$

The idea of adding other control conditions in BoostDream is inspired by ControlNet [Zhang *et al.*, 2023b], which is a text-to-image task solution allowing more input as task specific conditions like images. Its training goal can be written as:

$$\mathcal{L}_{\text{Control}} = \mathbb{E}_{x_0,t,c_t,c_f,\epsilon}\left[\|\epsilon - \epsilon_\theta(x_t, t, c_t, c_f)\|_2^2\right] \quad (4)$$

In this formulation, the terms c_t and c_f are task-specific conditions applied in ControlNet, like text-prompt and images.

3.2 3D Representation Initialization

Our method is applicable to a variety of 3D asset generation approaches based on differentiable rendering. In order to achieve this, we propose a generalized model distillation approach to fit a coarse 3D asset into a randomly initialized differential representation.

We define the coarse explicit 3D assets as a and the parameters of differentiable 3D representation as θ. In every iteration, the randomly sampled camera parameters c are passed into the renderer g to get the rendered images for these two 3D assets separately at the same view. With these pairs of images, we use L_1 loss function to make the randomly initialized 3D representation look as much like the coarse explicit 3D asset as possible in a very short time. The loss function is defined as:

$$L_1 = |g(a, c) - g(\theta, c)| \quad (5)$$

3.3 Multi-View Render System

To alleviate the Janus problem, we designed a new multi-view rendering method. As shown in the two boost stages of Figure 2, we initialize the default viewing direction in accordance with the input 3D model whose origin coincides with the origin of the coordinate system. At each iteration, we randomly sampled a camera position p_0 in spherical coordinates, consisting of elevation angle $\phi_{cam} \in [-10°, 70°]$, azimuth angle $\theta_{cam} \in [0°, 360°]$, field of view $f \in [0°, 180°]$ and camera distance d_{cam}. We further created a rotation axis, a random vector passing through the origin denoted as $\mathbf{a} = (a_x, a_y, a_z)$, a rotation angle denoted as α. To get multi-view camera position, a rotation matrix $\mathbf{R}$ is defined as:

$$\mathbf{R}(\mathbf{a}, \alpha) = \mathbf{I} + (\sin\alpha)\mathbf{K} + (1 - \cos\alpha)\mathbf{K}^2 \quad (6)$$

Where $\mathbf{I}$ is the identity matrix, and $\mathbf{K}$ is the skew-symmetric matrix generated from the rotation axis $\mathbf{a} = (a_x, a_y, a_z)$, defined as:

$$\mathbf{K} = \begin{pmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{pmatrix} \quad (7)$$

In our experimental settings, we define rotation angle $\alpha = 90°$, so the camera at position p_0 will rotate around axis $\mathbf{a}$ and get four camera positions. In each rotation iteration i, the camera position p_i after rotation follows:

$$p_i = \mathbf{R}(\mathbf{a}, i \times \alpha)p_0 \quad (8)$$

We use separate renderers for each task in different 3D representations. For a renderer g, we use the above four camera positions p_i along with other parameters to get camera parameters c_i. With a total of four sub-images generated from different camera parameters c_i and parameters of a 3D representation θ, stitch these four images together to create one large 2×2 composite image G. We also capture the normal maps rendered from 3D representation and follow a similar routine to get a large 2×2 normal map N.

$$G = \begin{pmatrix} g(\theta, c_0) & g(\theta, c_1) \\ g(\theta, c_2) & g(\theta, c_3) \end{pmatrix} \quad (9)$$

3.4 Multi-View SDS

The multi-view SDS loss function is designed to direct differentiable rendering models, utilizing both text prompt and multi-view conditions generated by the camera parameters we proposed in section 3.3 as the control guidance. We use multi-view normal maps here for multi-view conditions, as they provide a detailed depiction of surface normals, enhancing the capture of finer geometric and textural details. This enhancement is particularly beneficial for accurately preserving complex features like hair or surface irregularities, which are essential for lifelike 3D renderings.

The heart of this loss function is a new method of estimating noise guided by dual task-specific conditions, like normal maps and the text prompt. This dual-conditioned approach ensures that the generated content remains faithful to the surface details outlined by the normal maps while also aligning with the context provided by the text prompt. The noise estimation is formulated as:

$$\begin{aligned}\hat{\epsilon}_\phi(x_t; t, y, N) = \epsilon_\phi(x_t; t, y, \lambda * N) \\ + s * (\epsilon_\phi(x_t; t, y, \lambda * N) - \epsilon_\phi(x_t; t))\end{aligned} \quad (10)$$

Here, ϵ_ϕ denotes the noise predicted by the diffusion model, $\lambda \in [0, 1]$ balances the normal map conditioned and unconditioned noise predictions [Chen *et al.*, 2023c], and s is the scale of classifier-free guidance [Ho and Salimans, 2022]. N represents the multi-view normal maps captured following the rule we proposed in section 3.3, for normal map renderer g_N, we use the same camera parameters used by composite image G in Eq. 9:

$$N = \begin{pmatrix} g_N(\theta, c_0) & g_N(\theta, c_1) \\ g_N(\theta, c_2) & g_N(\theta, c_3) \end{pmatrix} \quad (11)$$

Like Eq. 3, the gradient of the multi-view SDS loss concerning the differentiable representations' parameters is defined as:

$$\nabla_\theta \mathcal{L}_{MV-SDS} = \mathbb{E}_{t,\epsilon}\left[w(t)(\hat{\epsilon}_\phi(x_t; t, y, N) - \epsilon)\frac{\partial G}{\partial \theta}\right] \quad (12)$$

Furthermore, We enhance the loss with orientation loss $\mathcal{L}_{\text{orient}}$ and opacity loss $\mathcal{L}_{\text{opacity}}$ proposed at [Poole *et al.*, 2022], when implement with NeRF.

The final loss function combines these components to ensure accurate surface detail representation, correct surface orientations, and optimal opacity:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{MV-SDS}(\phi, x) + \alpha\mathcal{L}_{\text{orient}} + \beta\mathcal{L}_{\text{opacity}} \quad (13)$$

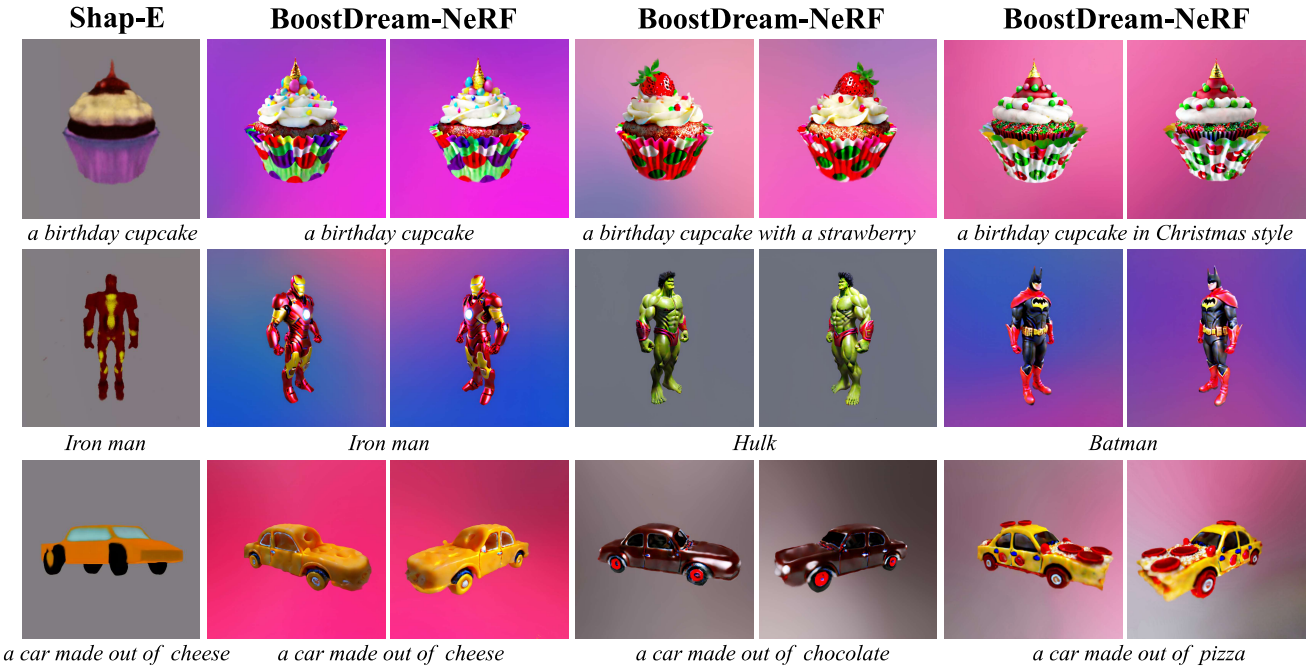


Figure 3: The first column is the Shap-E [Jun and Nichol, 2023a] results and the remaining column is the refined results of our method. The results show that BoostDream can refine and edit 3D assets according to different prompts based on input 3D assets.

where α and β are the weights for the orientation and opacity losses, respectively. This comprehensive loss function fosters the generation of 3D content that is both semantically and geometrically accurate, exhibiting realistic physical properties.

4 Experiments

In this section, we present a series of experiments designed to evaluate the capabilities of our BoostDream method. In Section 4.1, we detail the implementation specifics. Section 4.2 discusses the results of refinement experiments and explores its performance under various text prompts. In Section 4.3, we compare our method against existing techniques in terms of performance and speed. We conduct some ablation studies to illustrate the effect of each stage and condition setting of our method in Section 4.4. We also defined a user study to quantify the model performance which is shown in Section 4.5. These experiments are crucial in demonstrating the effectiveness of BoostDream in refining and editing 3D assets and the generalisability when applied to different 3D representations.

4.1 Implementation Details

All the experiments in this paper are conducted on a single NVIDIA V100 GPU with 32GB VRAM. We use Shap-E [Jun and Nichol, 2023a] from official implementation [Jun and Nichol, 2023b] to generate coarse 3D assets and fit them into differentiable 3D representations in the first 200 iterations. After initialization, we set 4800 iterations for the refining process, of which the first 1800 iterations are under the guidance of the coarse 3D assets, and the subsequent 3000 iterations are guided by the differentiable rendering result itself. We use ControlNet 1.1 with Stable Diffusion 1.5 [Zhang *et al.*, 2023b] as the diffusion model. In the comparison experiment, we carried out experiments on different differentiable 3D representations, including NeRF [Mildenhall *et al.*, 2021], DMTet [Shen *et al.*, 2021], and 3D Gaussian Splatting [Kerbl *et al.*, 2023]. Please refer to the Appendix [Yonghao *et al.*, 2024] for more details and results.

4.2 Refinement Experiment

We designed a refinement experiment that shows the powerful refining capabilities of our BoostDream. We choose Shap-E as the feed-forward generation method to generate coarse 3D assets. The coarse 3D assets will be sent to our refining method with the same text prompt. We use NeRF implementation here as an example. The result is shown in Figure 1. Our BoostDream can also refine coarse 3D assets with effective prompt-based editing. Experiments demonstrate its ability to refine assets using varied text prompts while retaining original features, as illustrated in Figure 3. Notably, examples like "Hulk" and "Batman" preserve the original posture and red elements from the "Iron Man" asset, showcasing a delicate balance between adaptation and preservation.

4.3 Comparison Experiment

Our BoostDream is a refining method that combines the advantages of the feed-forward generation method and the SDS-based optimization generation method. To demonstrate the effectiveness of our methods, we selected methods from each of the above two categories for the comparison experiment. For the feed-forward generation method, we choose Shap-E, which has an official implementation [Jun and Nichol, 2023b]. The 3D assets generated by Shap-E are also used

Figure 4: Comparision with Shap-E [Jun and Nichol, 2023a], DreamFusion [Poole *et al.*, 2022] and Magic3D [Lin *et al.*, 2023] for the same text-to-3D generation task. Our model has significantly stronger prompt relevancy and much better quality (best viewed by zooming in). See the results of our method on DMTet [Shen *et al.*, 2021] and 3D Gaussian Splatting [Kerbl *et al.*, 2023] in the Appendix [Yonghao *et al.*, 2024]
.

to initialize our BoostDream method. For the SDS-based optimization generation method, we choose DreamFusion and Magic3D, which are implemented by threestudio [Guo *et al.*, 2023] because no official code is available now. It is also worth noticing there are some differences between the original papers and the threestudio implementation. In this implementation, DreamFusion and Magcic3D use DeepFloyd [StabilityAI, 2023] as the diffusion model, while the originals use Imagen [Saharia *et al.*, 2022] and eDiff-I [Balaji *et al.*, 2022]. We use the default parameter of these methods. The experiment result is shown in Figure 4. Our method can generate higher-quality 3D assets compared to the above methods. In rows 1, 4, and 7, DreamFusion and Magic3D have arisen with the Janus problem, which means the Gundam has multiple arms, the horse has wrong legs, and the elephant has numerous heads, while our method can alleviate this problem due to the three-stage refine.

Also, compared to the two SDS-based optimization generation methods, the total generation time we need is reduced. Our method only requires 5000 iterations in total. As a comparison, the DreamFusion needs 15000 iterations to get a 3D asset, while the Magic3D needs 5000 iterations to train a coarse 3D asset and has a 3000 iteration refinement process. The speed comparison results are shown in Table 1. From the result, we can observe that apart from our high-quality generation result, BoostDream can rapidly finish optimization.

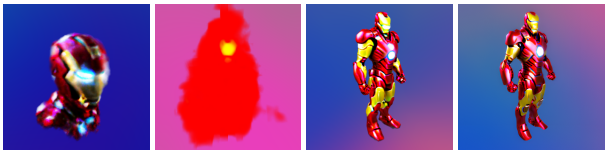
Method	Time (V100-sec)
BoostDream-NeRF	2038
BoostDream-DMTet	2128
BoostDream-3D Gaussian Splatting	1977
DreamFusion	3519
Magic3D	2355

Table 1: Speed comparison. Each time is the average training time of ten different text prompts. Note that DreamFusion and Magic3D are tested on threestudio implementation.

4.4 Ablation Study

In the proposed pipeline, we delineate three distinct stages: the initialization stage, the boost stage under the guidance of a coarse 3D asset, and the self-boost stage under the guidance of its intrinsic multiview. To elucidate the impact of each stage, we undertake an ablation study where we systematically eliminate each stage in isolation to assess their contributions. The results are shown in Figure 5, which illustrates that instead of using the initialization stage, the model may not converge quickly and have wrong guidance from the 3D assets we want when randomly initialized. Without coarse 3D asset guidance, the model demonstrates a propensity to evolve towards unforeseen outcomes. This phenomenon can be attributed to the premise that post-initialization errors perpetuate a cycle of learning based on these inaccuracies. Finally, without its self-guidance, the model is constrained by the original coarse 3D asset, thus failing to achieve a level of detail and quality that is otherwise attainable.

We also conducted two other ablation studies. One uses the coarse 3D asset generated by the feed-forward method to initialize the traditional SDS-based method. Another is using different control conditions to replace normal maps, such as canny edge and depth maps. See more ablation studies in the Appendix [Yonghao *et al.*, 2024].



(a) (b) (c) (d)

Figure 5: Ablation study. Fig(a) is without the initialization stage. Fig(b) is without the boost stage. Fig(c) is without the self-boost stage. Fig(d) is our complete BoostDream method.

4.5 User Study

Due to the lack of existing evaluation metrics for 3D generation, we conduct a user study to assess model performance. We create an evaluation set generated through 30 prompts using four methods. For a specific 3D asset, a rendered video with an input text prompt was shown to participants. 20 participants are invited to individually score each item in the set, focusing on prompt relevance and the quality of generated details, with scores ranging from 1 to 5. As shown in Figure 6, the results demonstrate that our method is markedly superior.

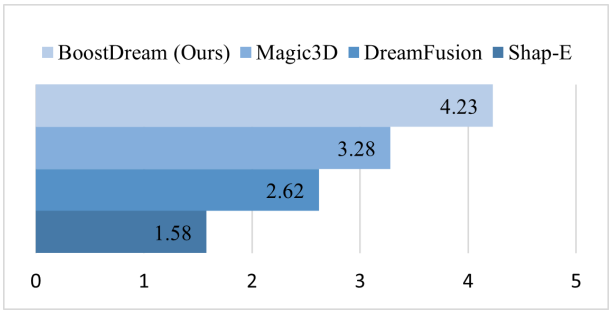


Figure 6: User study. Our model demonstrates a significant superiority in user preference, achieving high scores in a blinded survey with a maximum rating of 5. Here, we use BoostDream-NeRF as our implementation.

5 Conclusion

In summary, our research introduces BoostDream, a novel method that seamlessly combines differentiable rendering and text-to-image advancements to generate high-quality 3D assets efficiently. Central to our approach is the multi-view SDS, which effectively overcomes the Janus problem in generation processes. BoostDream can be applied to various differentiable 3D representations, generally improving the quality and reducing the time consumption of existing 3D generation methods. This work not only advances the quality and diversity of 3D assets but also sets a precedent for future innovations in 3D modeling, with wide-reaching implications for virtual reality and gaming industries.

Ethical Statement

The BoostDream proposed in this paper uses the 2D diffusion model as the prior to generating high-quality 3D assets with a multi-view strategy. We aim to reduce the cost of creating highly detailed 3D assets. However, we do note that the method could be applied to generate disinformation or violent and sexual content. With 2D diffusion as prior, it also inherits similar ethical and legal considerations to problematic biases and limitations that 2D diffusion models may have.

References

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

[Chen *et al.*, 2023a] Hansheng Chen, Jiatao Gu, Anpei Chen, Wei Tian, Zhuowen Tu, Lingjie Liu, and Hao Su. Single-stage diffusion nerf: A unified approach to 3d generation and reconstruction. *arXiv preprint arXiv:2304.06714*, 2023.

[Chen *et al.*, 2023b] 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.

[Chen *et al.*, 2023c] Yang Chen, Yingwei Pan, Yehao Li, Ting Yao, and Tao Mei. Control3d: Towards controllable text-to-3d generation. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 1148–1156, 2023.

[Deitke *et al.*, 2023] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. *arXiv preprint arXiv:2307.05663*, 2023.

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

[Gupta *et al.*, 2023] Anchit Gupta, Wenhan Xiong, Yixin Nie, Ian Jones, and Barlas Oğuz. 3dgen: Triplane latent diffusion for textured mesh generation. *arXiv preprint arXiv:2303.05371*, 2023.

[Ho and Salimans, 2022] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022.

[Ho *et al.*, 2020] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020.

[Jain *et al.*, 2022] 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*, pages 867–876, 2022.

[Jun and Nichol, 2023a] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023.

[Jun and Nichol, 2023b] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions. https://github.com/openai/shap-e, 2023.

[Kerbl *et al.*, 2023] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023.

[Lin *et al.*, 2023] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 300–309, 2023.

[Liu *et al.*, 2023a] Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Zexiang Xu, Hao Su, et al. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. *arXiv preprint arXiv:2306.16928*, 2023.

[Liu *et al.*, 2023b] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9298–9309, 2023.

[Liu *et al.*, 2023c] Zhen Liu, Yao Feng, Michael J Black, Derek Nowrouzezahrai, Liam Paull, and Weiyang Liu. Meshdiffusion: Score-based generative 3d mesh modeling. *arXiv preprint arXiv:2303.08133*, 2023.

[Lorraine *et al.*, 2023] Jonathan Lorraine, Kevin Xie, Xiaohui Zeng, Chen-Hsuan Lin, Towaki Takikawa, Nicholas Sharp, Tsung-Yi Lin, Ming-Yu Liu, Sanja Fidler, and James Lucas. Att3d: Amortized text-to-3d object synthesis. *arXiv preprint arXiv:2306.07349*, 2023.

[Melas-Kyriazi *et al.*, 2023] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. Realfusion: 360deg reconstruction of any object from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8446–8455, 2023.

[Metzer *et al.*, 2023] Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12663–12673, 2023.

[Mildenhall *et al.*, 2021] 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. *Communications of the ACM*, 65(1):99–106, 2021.

[Nichol *et al.*, 2022] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022.

[Ntavelis *et al.*, 2023] Evangelos Ntavelis, Aliaksandr Siarohin, Kyle Olszewski, Chaoyang Wang, Luc Van Gool, and Sergey Tulyakov. Autodecoding latent 3d diffusion models. *arXiv preprint arXiv:2307.05445*, 2023.

[Poole *et al.*, 2022] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

[Qian *et al.*, 2023] Guocheng Qian, Jinjie Mai, Abdullah Hamdi, Jian Ren, Aliaksandr Siarohin, Bing Li, Hsin-Ying Lee, Ivan Skorokhodov, Peter Wonka, Sergey Tulyakov, et al. Magic123: One image to high-quality 3d object generation using both 2d and 3d diffusion priors. *arXiv preprint arXiv:2306.17843*, 2023.

[Rombach *et al.*, 2022] 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*, pages 10684–10695, 2022.

[Saharia *et al.*, 2022] 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. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022.

[Sanghi *et al.*, 2023] Aditya Sanghi, Rao Fu, Vivian Liu, Karl DD Willis, Hooman Shayani, Amir H Khasahmadi, Srinath Sridhar, and Daniel Ritchie. Clip-sculptor: Zero-shot generation of high-fidelity and diverse shapes from natural language. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18339–18348, 2023.

[Schuhmann *et al.*, 2022] 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. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022.

[Shen *et al.*, 2021] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[Shi *et al.*, 2023] Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation. *arXiv preprint arXiv:2308.16512*, 2023.

[StabilityAI, 2023] StabilityAI. Deepfloyd. https://huggingface.co/DeepFloyd, 2023.

[Tang *et al.*, 2023] Shitao Tang, Fuyang Zhang, Jiacheng Chen, Peng Wang, and Yasutaka Furukawa. Mvdiffusion: Enabling holistic multi-view image generation with correspondence-aware diffusion. *arXiv preprint arXiv:2307.01097*, 2023.

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

[Wang *et al.*, 2023a] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12619–12629, 2023.

[Wang *et al.*, 2023b] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *arXiv preprint arXiv:2305.16213*, 2023.

[Yonghao *et al.*, 2024] Yu Yonghao, Zhu Shunan, Qin Huai, and Li Haorui. Boostdream: Efficient refining for high-quality text-to-3d generation from multi-view diffusion. https://boostdream.github.io/, 2024.

[Zeng *et al.*, 2022] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. *arXiv preprint arXiv:2210.06978*, 2022.

[Zhang *et al.*, 2023a] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.

[Zhang *et al.*, 2023b] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. https://huggingface.co/lllyasviel, 2023.

[Zhao *et al.*, 2023] Minda Zhao, Chaoyi Zhao, Xinyue Liang, Lincheng Li, Zeng Zhao, Zhipeng Hu, Changjie Fan, and Xin Yu. Efficientdreamer: High-fidelity and robust 3d creation via orthogonal-view diffusion prior. *arXiv preprint arXiv:2308.13223*, 2023.

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