

# Supplementary Materials for PlacidDreamer

Anonymous Authors

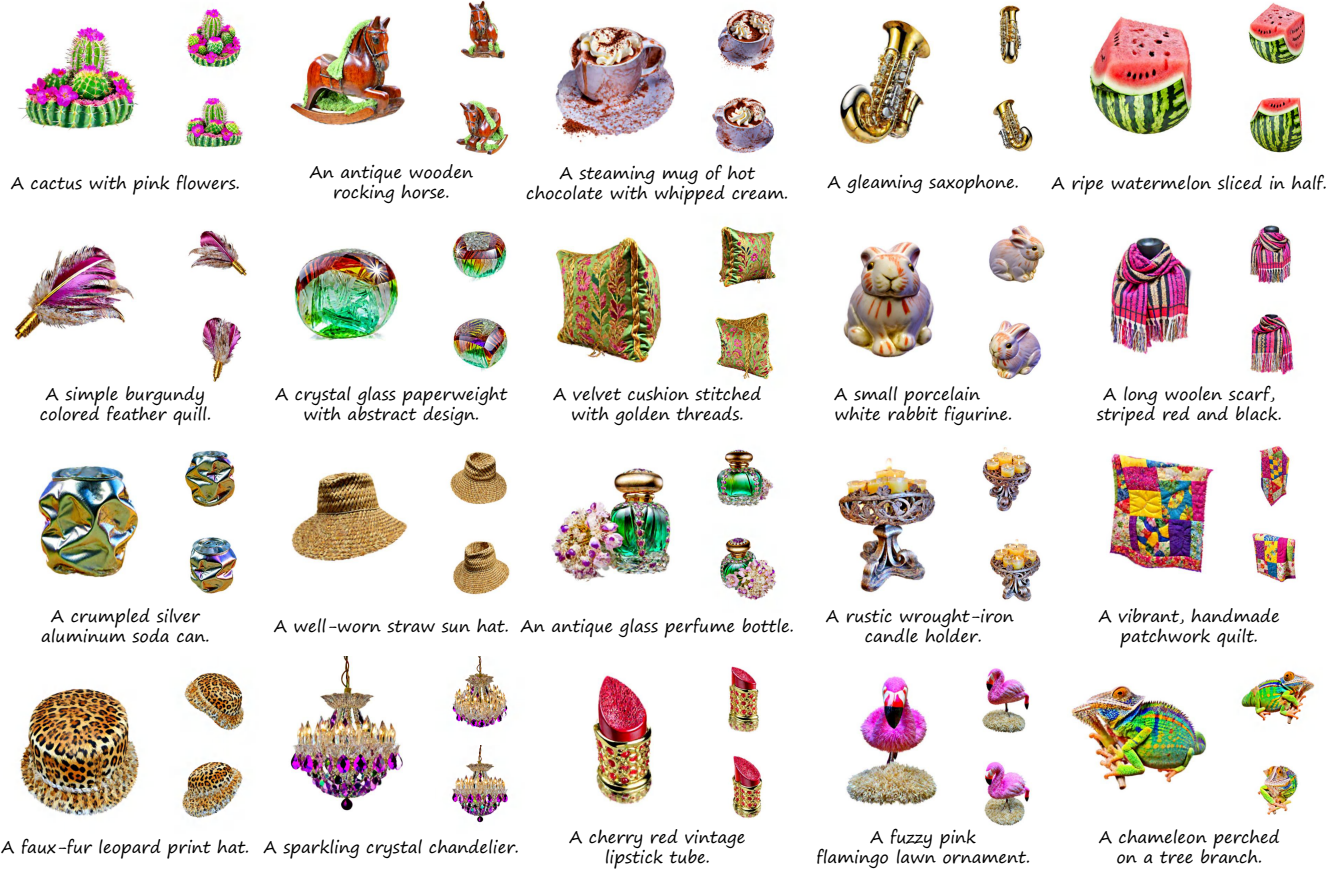


Figure 1: More 3D generation results of PlacidDreamer.

## 1 DEMO VIDEO

We provide a demo video for a direct and precise demonstration of:

- Visualization of 2D generation experiments using Balanced Score Distillation (BSD);
- Comparative performance visualization of PlacidDreamer against baseline 3D Gaussian methods;
- A visual gallery of the 3D generation results of PlacidDreamer.

## 2 MORE RESULTS OF PLACIDDREAMER

We provide more 3D generation results of PlacidDreamer in Figure 1.

## 3 EXPERIMENTS OF BSD ALGORITHM

BSD is a versatile score distillation algorithm. We demonstrate its stability and versatility by applying it across various text-to-3D open-source frameworks. Initially, within the general framework of ThreeStudio [2], we compare the generation capabilities of SDS [6], CSD [11], VSD [10], and BSD for NeRF [5] and DM-Tet [8].

Subsequently, we substitute the score distillation algorithms in different frameworks with BSD, ensuring that all other parameters in the experiments remains constant, to validate its enhancements.

As illustrated in Figure 2, we present the experimental results on ThreeStudio. To ensure a fair comparison, we maintain all parameters that are unrelated to score distillation at constant values, while meticulously adjusting hyper-parameters of each score distillation algorithm to achieve optimal effects. In the generation of NeRFs, SDS exhibits the least detail levels and suffers from unrealistic colors. CSD, comparing with SDS, captures more pronounced details. However, the over-saturation curtails its realism. The performances of VSD and BSD are closely matched, with VSD displaying finer details and BSD displaying more accurate color distributions. BSD stands out in color accuracy, nearly matches VSD in detail richness, and equals SDS in speed, thus positioning BSD as the superior choice over previous score distillation methods. For DM-Tet refinement, we use a NeRF sample generated by BSD as mesh initialization. We observe that these algorithms exhibit similar quality with SDS and CSD having subtle color deviations.

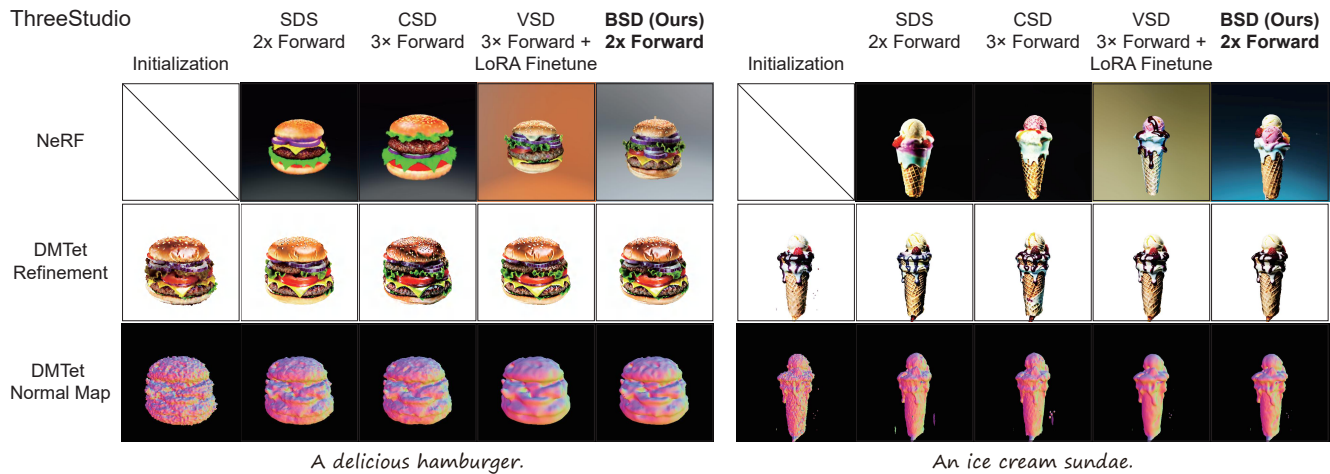


Figure 2: Results on ThreeStudio comparing score distillation algorithms for basic NeRF generation and DMTet refinement.

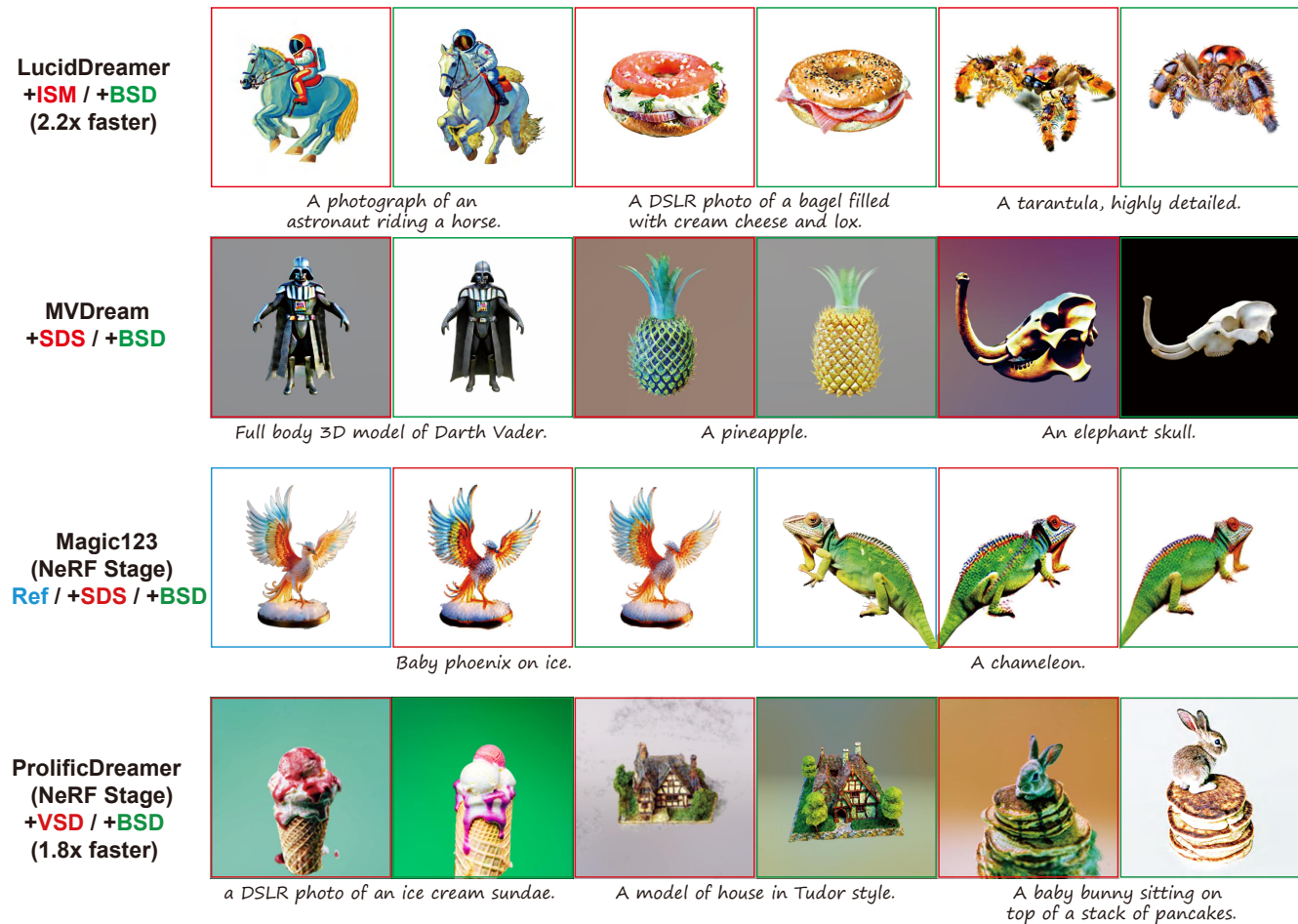
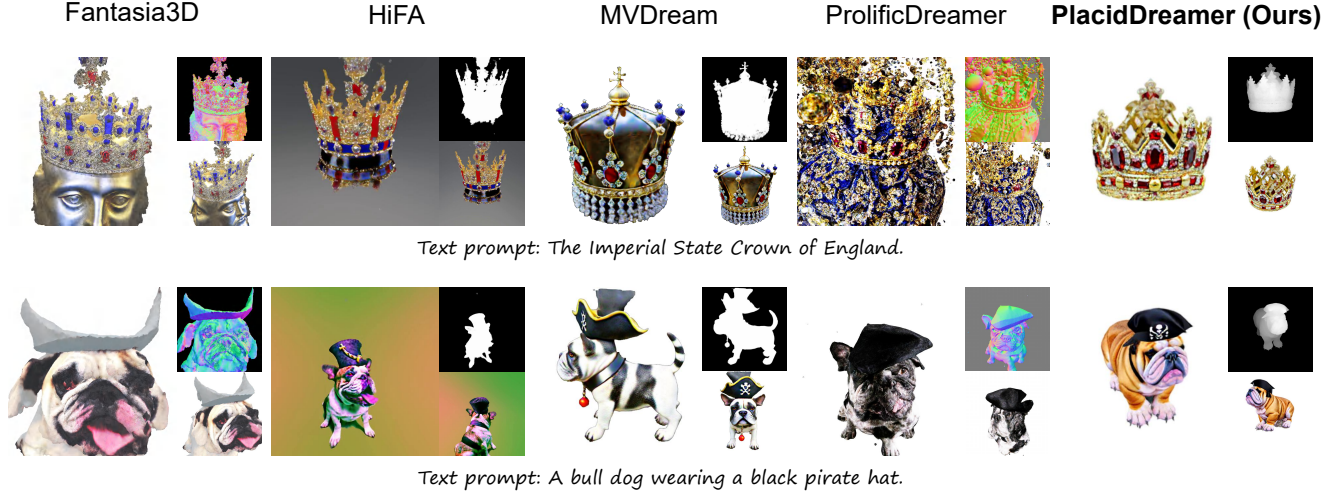


Figure 3: Results of replacing other score distillation methods with BSD in text-to-3D pipelines.



**Figure 4: Qualitative comparison with NeRF-based baseline methods.**

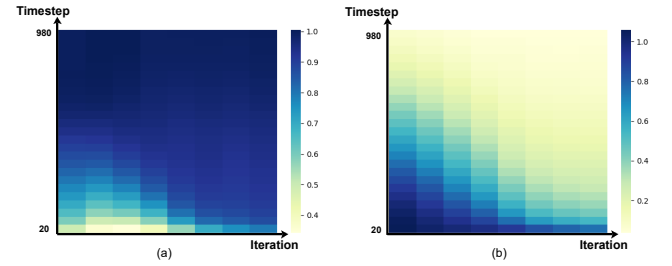
As shown in Figure 3, replacing previous score distillation algorithms with the BSD algorithm consistently improves performance across various frameworks. In the LucidDreamer [3] pipeline, which employs Interval Score Matching (ISM), the BSD algorithm not only trains 2.2 times faster but also significantly enhances semantic alignment and texture fidelity. ISM prioritizes guidance at lower timesteps by assigning greater weights. Without resolving the conflict between classifier and smoothing guidance, this leads to over-saturation in certain scenarios (e.g., the astronaut case). Moreover, optimizations at low timesteps often add irrelevant details, enhancing visual complexity but sometimes causing semantic inconsistencies with the text prompt (e.g., the bagel case). In the MVDream [9] pipeline with the SDS algorithm, BSD effectively reduces the color distortion problems. The capability of BSD to reduce over-saturation is further demonstrated in experiments conducted on the Magic123 [7] pipeline, which also show its applicability to multi-view diffusion models [4]. Furthermore, experiments on ProlificDreamer [10] reveal that BSD not only runs much faster but also matches VSD in detail level. The experiments validate BSD as a versatile and robust choice for score distillation.

#### 4 COMPARISON WITH MORE BASELINE METHODS

We conduct qualitative comparisons with several NeRF-based [5] baseline methods, including Fantasia3D [1], HiFA [12], MVDream [9], and ProlificDreamer [10]. As shown in Figure 4, we present the results of these methods generating responses to the same prompts. Fantasia3D utilizes a unique geometry generation process for enhanced geometry generation, yet it does not match the overall quality of PlacidDreamer. MVDream is capable of generating stable, multi-view consistent 3D models. However, its training process reduces resolution, resulting in the loss of high-frequency details. ProlificDreamer produces meshes with high-fidelity textures, but it sometimes fails to converge, suffers from severe multi-face problems, and incurs a time cost significantly exceeding other methods. Our PlacidDreamer, with improvements in BSD and pipeline design,

achieves balanced saturation, refined details, and more stable generation. Therefore, compared with previous NeRF-based methods, PlacidDreamer is capable of generating higher-quality 3D assets.

#### 5 RELATIONSHIP WITH PREVIOUS METHODS



**Figure 5: Visualization of the Euclidean norm of smoothing guidance during training: (a)  $\delta_{SG} = \epsilon(\mathbf{x}_t, t, \emptyset)$  of BSD. (b)  $\delta'_{SG} = \epsilon(\mathbf{x}_t, t, \emptyset) - \epsilon$  of SDS.**

**Comparison with SDS.** The main differences between our approach (BSD) and SDS lie in two aspects. The first is the incorporation of the Multiple-Gradient Descent Algorithm (MGDA), as elaborated in the main text. The second is the omission of the final term  $-\epsilon$ . The formula we derived is similar to the one provided in the appendix of the DreamFusion [6] paper. The appendix claims that introducing  $-\epsilon$  helps to reduce high variance of the gradients. Despite the common inclusion of  $-\epsilon$  in most previous works that follow the SDS paradigm, we have observed that  $-\epsilon$  can be omitted for three reasons.

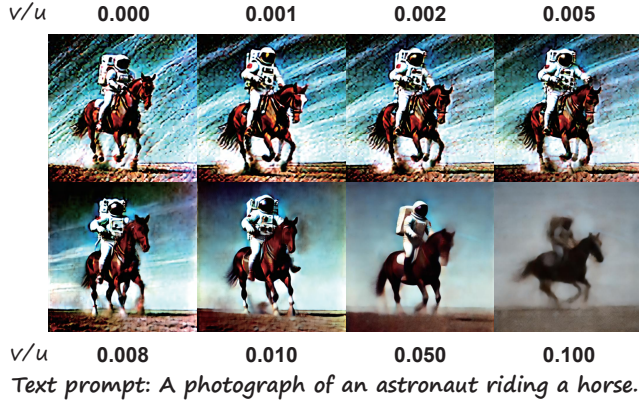
Firstly, the introduction of  $-\epsilon$  causes the magnitude of  $\delta'_{SG} = \epsilon(\mathbf{x}_t, t, \emptyset) - \epsilon$  exhibit greater variance across different timesteps compared with  $\delta_{SG} = \epsilon(\mathbf{x}_t, t, \emptyset)$ . As we can clearly see in Figure 5, the color blocks within each column of  $\delta_{SG}$  are almost identical, indicating that their magnitudes are similar. In contrast, the colors



in each column of  $\delta'_{SG}$  change significantly with the timestep, which indicates that their magnitudes vary widely. This results in a fixed CFG value being more difficult to control balance. Additionally, we find that  $\delta'_{SG}$  has a higher likelihood of producing the obtuse angles between two optimization directions at low timesteps. These factors result in a more severe over-saturation problem.

Secondly, the introduction of  $-\epsilon$  causes the magnitude of  $\delta'_{SG}$  to decrease during optimization, disrupting the balance. As the rendered 2D images progressively mirror the distribution of real 2D images, the predicted noise  $\epsilon(\mathbf{x}_t, t, \emptyset)$  and the added random noise  $\epsilon$  become numerically correlated. This correlation lowers the magnitude of  $\delta'_{SG}$ , visibly lightening the color blocks in each row in Figure 5 (b). Consequently, in this scenario, classifier guidance is likely to dominate when the CFG parameter is fixed, leading to a more pronounced over-saturation issue. When MGDA is applied, its mathematical property—increasing the proportion of components as their magnitudes decrease—causes  $\delta'_{SG}$  to likely dominate in the MGDA algorithm, exacerbating the over-smoothing problem. Thus, the magnitude reductions caused by  $-\epsilon$  are detrimental to achieving balance.

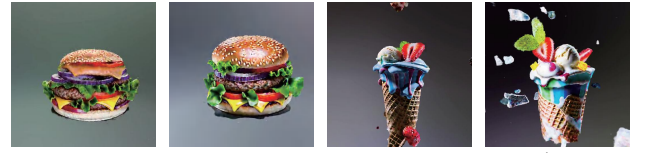
Thirdly, despite  $-\epsilon$  having a mathematical expectation of zero, practical challenges arise. Text-to-3D algorithms typically run only a few thousand to tens of thousands of iterations. With 1000 different timesteps involved, the few dozen random samples per timestep are insufficient to mitigate their mutual impacts effectively.



**Figure 6: The influence of the ratio between smoothing guidance and classifier guidance, represented as  $v/u$  ( $-\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|y) = u \cdot \delta_{CG} + v \cdot \delta_{SG}$ ). When the ratio  $v/u$  is set to zero, it corresponds to CSD.**

**Comparison with CSD.** The decomposition method of CSD is similar to ours, as both include a classifier guidance term  $\delta_{CG}$ . However, their  $\delta^{\text{gen}} = \epsilon(\mathbf{x}_t, t, y) - \epsilon$ , whereas our  $\delta_{SG} = \epsilon(\mathbf{x}_t, t, \emptyset)$ . In most cases, experimental results using  $\epsilon(\mathbf{x}_t, t, y)$  and  $\epsilon(\mathbf{x}_t, t, \emptyset)$  do not differ significantly. It should be noted that at the initial stages of 3D generation, when initialized to a sphere, the image is smooth, thus  $\epsilon(\mathbf{x}_t, t, \emptyset)$  is correlated with  $\epsilon$ , while  $\epsilon(\mathbf{x}_t, t, y)$  is not correlated with  $\epsilon$ . This means that the  $\epsilon(\mathbf{x}_t, t, \emptyset)$  is more closely related to smoothness. Therefore, it is more accurate to refer to our decomposition as the 'smoothing guidance'.

A more significant difference arises in understanding the effects of the decomposed terms. CSD discovers that  $\delta^{\text{gen}}$  alone does not function effectively and occupies a very low proportion in terms of magnitude, leading to the conclusion that  $\delta^{\text{gen}}$  can be discarded. However, according to our modeling, using only  $\delta_{CG}$  causes the generated results to overfit to a mode learned by a classifier, thereby distilling discriminative power but not utilizing generative capabilities. This leads to noticeable artifacts in 2D experiments and over-saturation issues in 3D experiments produced by CSD. We display the results of the score distillation decomposition experiments in higher detail in Figure 6. Furthermore, we believe that precisely because the proportion of  $\delta_{SG}$  is very small, the MGDA algorithm is necessary to ensure its effects are not overwhelmed, hence we propose the BSD algorithm.



**Figure 7: The results of applying MGDA to VSD.**

**Comparison with VSD.** The derivation approach for VSD involves sampling from a 3D distribution as part of variational inference and utilizing a particle-based ODE (Ordinary Differential Equation) solver. The final expression derived is:

$$\nabla_{\theta} \mathcal{L}_{\text{VSD}}(\theta) = \mathbb{E}_{t, \epsilon, c} [\omega(t) (\epsilon_{\text{pretrain}}(\mathbf{x}_t, t, y) - \epsilon_{\phi}(\mathbf{x}_t, t, y)) \frac{\partial g(\theta, \pi)}{\partial \theta}],$$

where  $\theta$  represents the parameters of the 3D model,  $\epsilon_{\text{pretrain}}$  is the noise predicted by the pre-trained diffusion model,  $\epsilon_{\phi}$  is the noise predicted by the diffusion model fine-tuned on the optimizing 3D assets, and  $g(\theta, \pi)$  is the differentiable renderer for the 3D assets from perspective  $\pi$ .

Unlike SDS, VSD cannot be decomposed into a combination of classifier guidance and smoothing guidance, because  $\epsilon_{\phi}(\mathbf{x}_t, t, y)$  inherently carries probabilistic meanings, complicating its integration with any guidance term. However, we still test the effects of incorporating  $-\epsilon_{\phi}(\mathbf{x}_t, t, y)$  into the smoothing guidance and applying MGDA. The results, as illustrated in Figure 7, demonstrate improvements in detail level and content richness. Since VSD modeling is not the central focus of our paper, we leave the potential exploration of integrating VSD with MGDA to future work.

## 6 IMPLEMENTATION DETAILS

In the Latent-Plane module, we utilize two Multi-Head Self-Attention layers, each with eight heads and a feature dimension of 32, to extract sigma features. An additional two layers are used to enhance the multi-view features. For embeddings not derived from neural networks, we employ sinusoidal encoding. To minimize computational demands, all MLP networks are comprised of a single linear layer. During LoRA finetuning, only the UNet LoRA layers are finetuned at a learning rate of  $1 \times 10^{-4}$  across 400 iterations. In our BSD implementation, we set  $\lambda = 25$  and  $\omega(t) \propto \alpha_t^2$  to achieve an optimal balance of rich detail and accurate color reproduction.

REFERENCES

[1] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. 2023. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. *arXiv preprint arXiv:2303.13873* (2023).

[2] 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. 2023. threestudio: A unified framework for 3D content generation. <https://github.com/threestudio-project/threestudio>.

[3] Yixun Liang, Xin Yang, Jiantao Lin, Haodong Li, Xiaogang Xu, and Yingcong Chen. 2023. LucidDreamer: Towards High-Fidelity Text-to-3D Generation via Interval Score Matching. *arXiv:2311.11284 [cs.CV]*

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

[5] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.

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

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

[8] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. 2021. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. *Advances in Neural Information Processing Systems* 34 (2021), 6087–6101.

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

[10] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. 2023. ProlificDreamer: High-Fidelity and Diverse Text-to-3D Generation with Variational Score Distillation. *arXiv preprint arXiv:2305.16213* (2023).

[11] Xin Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Song-Hai Zhang, and Xiaojuan Qi. 2023. Text-to-3d with classifier score distillation. *arXiv preprint arXiv:2310.19415* (2023).

[12] Joseph Zhu and Peiye Zhuang. 2023. HiFA: High-fidelity Text-to-3D with Advanced Diffusion Guidance. *arXiv preprint arXiv:2305.18766* (2023).