SteinDreamer: Variance Reduction for Text-to-3D Score Distillation via Stein Identity

Peihao Wang[1], Zhiwen Fan[1], Dejia Xu[1], Dilin Wang[2], Sreyas Mohan[2], Forrest Iandola[2], Rakesh Ranjan[2], Yilei Li[2], Qiang Liu[1], Zhangyang Wang[1], Vikas Chandra[2]

[1]University of Texas at Austin, [2]Meta Reality Labs

vita-group.github.io/SteinDreamer/

Abstract

Score distillation has emerged as one of the most prevalent approaches for text-to-3D asset synthesis. Essentially, score distillation updates 3D parameters by lifting and back-propagating scores averaged over different views. In this paper, we reveal that the gradient estimation in score distillation is inherent to high variance. Through the lens of variance reduction, the effectiveness of SDS and VSD can be interpreted as applications of various control variates to the Monte Carlo estimator of the distilled score. Motivated by this rethinking and based on Stein's identity, we propose a more general solution to reduce variance for score distillation, termed *Stein Score Distillation (SSD)*. SSD incorporates control variates constructed by Stein identity, allowing for arbitrary baseline functions. This enables us to include flexible guidance priors and network architectures to explicitly optimize for variance reduction. In our experiments, the overall pipeline, dubbed *SteinDreamer*, is implemented by instantiating the control variate with a monocular depth estimator. The results show that SSD can effectively reduce the distillation variance and consistently improve visual quality for both object- and scene-level generation.

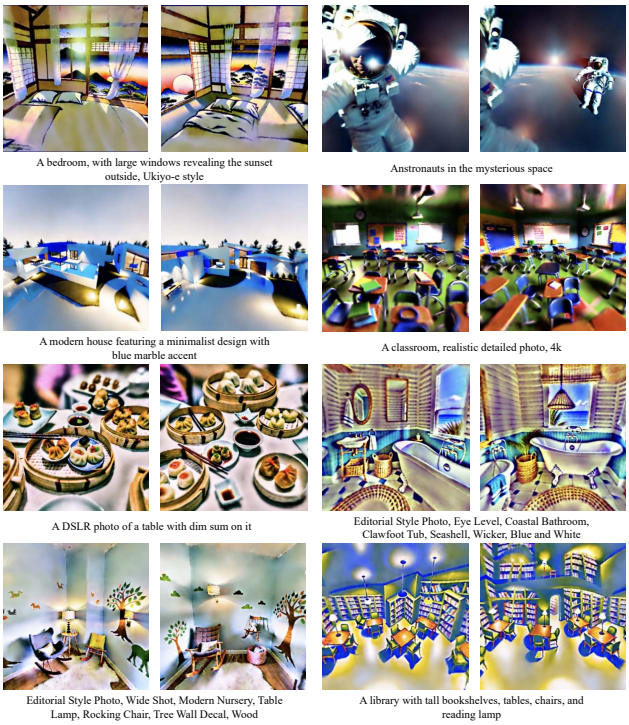

Figure 1: **A gallery of text-to-3D results generated by SteinDreamer.** Our method can synthesize large-scale scenes with smooth geometries and rich textures according to complex text prompts. Zoom in for the best view.

1 Introduction

There have been recent significant advancements in text-to-image generation, driven by diffusion models. Notable examples include Nichol et al. (2021); Ramesh et al. (2021, 2022) and Sohl-Dickstein et al. (2015); Ho et al. (2020); Song and Ermon (2019); Song et al. (2020); Dhariwal and Nichol (2021). These developments have sparked growing interest in the realm of text-guided 3D generation. This emerging field aims to automate and accelerate 3D asset creation in the applications of virtual reality, movies, and gaming. However, 3D synthesis poses significantly greater challenges. Directly training generative models using 3D data, as explored in works by (Wu et al., 2016; Yang et al., 2019; Cai et al., 2020; Nichol et al., 2022; Jun and Nichol, 2023; Chan et al., 2022; Shue et al., 2022), faces practical hurdles due to the scarcity of high-quality and diverse

Proceedings of the 28[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

data. Moreover, the inherent complexity of generative modeling with 3D representations adds an extra layer of intricacy to this endeavor.

In recent times, techniques based on score distillation (Poole et al., 2022; Wang et al., 2023c), exemplified by DreamFusion and ProlificDreamer, have gained prominence. These methods have garnered attention for their ability to effectively bypass the need for 3D data by leveraging a 2D diffusion model for 3D generation. In particular, Poole et al. (2022) introduces Score Distillation Sampling (SDS), which optimizes a differentiable 3D representation, such as NeRF (Mildenhall et al., 2020), by lifting and back-propagating image scores from a pre-trained text-to-image diffusion model. Among its subsequent works (Lin et al., 2023; Wang et al., 2023a; Chen et al., 2023a; Metzer et al., 2023; Wang et al., 2023b), ProlificDreamer stands out for significantly enhancing the generation quality through derived Variational Score Distillation (VSD) (Wang et al., 2023c). In particular, VSD introduces an additional score for rendered image distribution that enhances the quality of distillation results.

Despite all these progresses, it is widely recognized that gradient obtained through score distillation techniques tend to be noisy and unstable due to the high uncertainty in the denoising process and the small batch size limited by computational constraints. Consequently, this leads to slow convergence and suboptimal solutions. In this paper, we address this issue by proposing a unified variance reduction approach. We reveal that both the noise term in SDS and the extra score function introduced by VSD have zero means, and thus can be regarded as *control variates*. The update of VSD is equivalent to the update of SSD in expectation. However, the gradient variance is smaller in VSD due to a better choice of the control variate.

Building on these insights, we present a more flexible control variate for score distillation, leveraging Stein identity (Stein, 1972; Chen, 1975; Gorham and Mackey, 2015), dubbed *Stein Score Distillation (SSD)*. Stein's identity, given by $\mathbb{E}_{\boldsymbol{x}\sim p}[\nabla \log p(\boldsymbol{x}) \cdot \phi(\boldsymbol{x})^\top + \nabla_{\boldsymbol{x}} \phi(\boldsymbol{x})] = 0$ for any distribution p and function ϕ satisfying mild regularity conditions (Stein, 1972; Gorham and Mackey, 2015; Liu et al., 2016). This formulation establishes a broader class of control variates due to its zero means, providing flexibility in optimizing function ϕ for variance reduction. Specifically, our SSD frames the distillation update as a combination of the score estimation from a pre-trained diffusion model and a control variate derived from Stein's identity. The first term aligns with that in SDS and VSD, serving to maximize the likelihood of the rendered image. The second control variate is tailored to specifically reduce gradient variance. Importantly, our construction allows us to incorporate arbitrary prior knowledge and network architectures in ϕ, facilitating the design of control variates highly correlated with the lifted image score, leading to a significant reduction in gradient variance.

We integrate our proposed SSD into a text-to-3D generation pipeline, coined as *SteinDreamer*. Through extensive experiments, we demonstrate that SteinDreamer can consistently mitigate variance issues within the score distillation process. For both 3D object and scene-level generation, SteinDreamer outperforms DreamFusion and ProlificDreamer by providing detailed textures, precise geometries, and effective alleviation of the Janus (Hong et al., 2023) and ghostly (Warburg et al., 2023) artifacts (see Fig. 1). Last but not least, SteinDreamer, with the reduced variance, accelerates the convergence of 3D generation process by 14%-22%.

2 Preliminaries

We briefly summarize the major components used in the text-to-3D pipeline with score distillation, while introducing our notations along the way.

2.1 Score Distillation

Diffusion models, as demonstrated by various works (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song and Ermon, 2019; Song et al., 2020), have proven to be highly effective in text-to-image generation. Diffusion models learn a series of score functions $\nabla \log p_t(\boldsymbol{x}_t|\boldsymbol{y})$ for Gaussian perturbed image distribution $p_t(\boldsymbol{x}_t|\boldsymbol{y}) = \int \mathcal{N}(\boldsymbol{x}_t|\alpha_t \boldsymbol{x}_0, \sigma_t^2 \boldsymbol{I}) p_0(\boldsymbol{x}_0|\boldsymbol{y}) d\boldsymbol{x}_0$, where $\alpha_t, \sigma_t > 0$ are annealing noise coefficients, and $\boldsymbol{y}$ are text embeddings.

Build upon the success of 2D diffusion models, Poole et al. (2022); Wang et al. (2023a); Lin et al. (2023); Chen et al. (2023a); Tsalicoglou et al. (2023); Metzer et al. (2023); Wang et al. (2023c); Huang et al. (2023) demonstrate the feasibility of using a 2D generative model to create 3D asserts. Among these works, score distillation techniques play a central role by providing a way to guide a differentiable 3D representation using a pre-trained text-to-image diffusion model. Essentially, score distillation lifts and back-propagates signals estimated from a 2D prior to update a differentiable 3D representation, such as Neural Radiance Field (NeRF) (Mildenhall et al., 2020), via the chain rule (Wang et al., 2023a). There are primarily two types of distillation schemes elaborated below:

Score Distillation Sampling. The main idea of *Score Distillation Sampling (SDS)* is to utilize a score function of some pre-trained 2D image distribution $\nabla \log p_t$ and the denoising score matching loss to optimize a 3D representation, denoted as $\boldsymbol{\theta}$, such that

it semantically matches a given text prompt based on its multi-view projections. By taking derivatives with respect to 3D parameters $\boldsymbol{\theta}$ and dropping the Jacobian matrix of the score function, SDS yields the following update rule [1]:

$$\boldsymbol{\Delta}_{SDS} \triangleq \mathbb{E}\left[\omega(t)\frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial \boldsymbol{\theta}}\left(\sigma_t \nabla \log p_t(\boldsymbol{x}_t|\boldsymbol{y}) - \boldsymbol{\epsilon}\right)\right], \tag{1}$$

where the expectation is taken over time step $t \sim \mathcal{U}[0,T]$, camera pose $\boldsymbol{c} \sim p_c$, and white noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0},\boldsymbol{I})$. The noisy input of score function is denoted as $\boldsymbol{x}_t = \alpha_t g(\boldsymbol{\theta},\boldsymbol{c}) + \sigma_t \boldsymbol{\epsilon}$ and $\omega(t) > 0$ are time-dependent coefficients. $g(\boldsymbol{\theta},\boldsymbol{c})$ renders a 2D view from $\boldsymbol{\theta}$ given $\boldsymbol{c}$. In this work, we represent $\boldsymbol{\theta}$ as a NeRF, wherein $g(\boldsymbol{\theta},\boldsymbol{c})$ represents a volumetric renderer displaying each image pixel under camera pose $\boldsymbol{c}$ by performing ray tracing based rendering (Max, 1995). Meanwhile, $-\sigma_t \nabla \log p_t$ can be surrogated by a noise estimator from a pre-trained diffusion model.

Variational Score Distillation. ProlificDreamer introduced a new variant of score distillation, *Variational Score Distillation (VSD)* (Wang et al., 2023c), through the lens of particle-based variational inference (Liu and Wang, 2016; Liu, 2017; Detommaso et al., 2018). ProlificDreamer minimizes the KL divergence between $p_t(\boldsymbol{x})$ and the image distribution rendered from a 3D representation $\boldsymbol{\theta}$. The authors derive the following update rule through Wasserstein gradient flow:

$$\boldsymbol{\Delta}_{VSD} \triangleq \mathbb{E}\left[\omega(t)\frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial \boldsymbol{\theta}}(\sigma_t \nabla \log p_t(\boldsymbol{x}_t|\boldsymbol{y}) - \sigma_t \nabla \log q_t(\boldsymbol{x}_t|\boldsymbol{c}))\right], \tag{2}$$

where the expectation is taken over all relevant variables same with Eq. 1. Notably, there emerges a new score function of probability density function $q_t(\boldsymbol{x}|\boldsymbol{c})$, which characterizes the conditional distribution of noisy rendered images given the camera pose $\boldsymbol{c}$. While $\nabla \log p_t$ can be approximated in a similar manner using an off-the-shelf diffusion model, $\nabla \log q_t$ is not readily available. The solution provided by Wang et al. (2023c) is to fine-tune a pre-trained diffusion model using the rendered images. The approach results an alternating optimization paradigm between objective Eq. 2 and additional score matching loss: $\min_{\nabla \log q_t} \mathbb{E}_{t,\boldsymbol{c},\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})}\left[\omega(t)\|\sigma_t \nabla \log q_t(\boldsymbol{x}_t|\boldsymbol{c}) - \boldsymbol{\epsilon}\|_2^2\right]$, where $\nabla \log q_t(\boldsymbol{x}_t|\boldsymbol{c})$ is a diffusion model initialized with the pre-trained $\nabla \log p_t(\boldsymbol{x}_t|\boldsymbol{y})$, parameterized by LoRA (Hu et al., 2021), and additionally conditioned on camera pose.

[1] By default, we use transposed Jacobian matrices, i.e., $\left[\frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial \boldsymbol{\theta}}\right]_{i,j} = \frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})_j}{\partial \boldsymbol{\theta}_i}$.

2.2 Control Variate

Later in this work, we will introduce control variate into the context of score distillation. Control variate is a widely utilized technique to reduce variance for Monte Carlo estimator in various fields, including physical simulation (Davies et al., 2004), graphical rendering (Kajiya, 1986; Müller et al., 2020), network science (Meyn, 2008; Chen et al., 2017), and reinforcement learning (Williams, 1992; Sutton et al., 1998, 1999; Liu et al., 2017). Suppose we want to estimate the expectation $\mathbb{E}_{\boldsymbol{x}\sim q(\boldsymbol{x})}[f(\boldsymbol{x})]$ for some function f via Monte Carlo samples $\{\boldsymbol{x}_i \in \mathbb{R}^D\}_{i=1}^N$: $\boldsymbol{\Delta} = \frac{1}{N}\sum_{i=1}^N f(\boldsymbol{x}_i)$. The estimator $\boldsymbol{\Delta}$ is supposed to have large variance when N is small. Consider we have control variate as a function h with zero mean under $q(\boldsymbol{x})$. Then we can construct an unbiased estimator by adding term $\boldsymbol{\xi} = \frac{1}{N}\sum_{i=1}^N h(\boldsymbol{x}_i)$: $\boldsymbol{\Delta}^\dagger = \frac{1}{N}\sum_{i=1}^N (f(\boldsymbol{x}_i) + \boldsymbol{\mu} \odot h(\boldsymbol{x}_i))$, where $\boldsymbol{u} \in \mathbb{R}^D$ is a group of reweighting coefficients and $\odot$ denotes element-wise multiplication. The resultant estimator has variance for the i-th entry: $\mathbb{V}[\boldsymbol{\Delta}_i^\dagger] = \mathbb{V}[\boldsymbol{\Delta}_i] + \boldsymbol{\mu}_i^2 \mathbb{V}[\boldsymbol{\xi}_i] + 2\boldsymbol{\mu}_i \mathbb{E}[\boldsymbol{\Delta}\boldsymbol{\xi}^\top]_{ii}$. It is possible to reduce $\mathbb{V}[\boldsymbol{\Delta}_i^\dagger]$ by selecting h and $\boldsymbol{u}$ properly. To maximize variance reduction, $\boldsymbol{u}$ is chosen as $-\mathbb{E}[\boldsymbol{\Delta}\boldsymbol{\xi}^\top]_{ii}/\mathbb{V}[\boldsymbol{\xi}_i]$, leading to $\mathbb{V}[\boldsymbol{\Delta}_i^\dagger] = (1 - \mathrm{Corr}(\boldsymbol{\Delta}_i, \boldsymbol{\xi}_i)^2)\mathbb{V}[\boldsymbol{\Delta}_i]$, where $\mathrm{Corr}(\cdot,\cdot)$ denotes the correlation coefficient. This signifies that higher correlation between functions f and h leads to higher variance reduction.

3 Rethinking SDS and VSD: A Control Variate Perspective

In this section, we reveal that the variance of update estimation may play a key role in score distillation. At first glance, SDS and VSD differ in their formulation and implementation. However, our first theoretical finding reveals that SDS and (single-particle) VSD are equivalent in their expectation, i.e., $\boldsymbol{\Delta}_{SDS} = \boldsymbol{\Delta}_{VSD}$. We formally illustrate this observation below.

As a warm-up, we inspect SDS by decoupling the score function and noise terms:

$$\boldsymbol{\Delta}_{SDS} = \mathbb{E}\underbrace{\left[\omega(t)\frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial \boldsymbol{\theta}}\sigma_t \nabla \log p_t(\boldsymbol{x}_t|\boldsymbol{y})\right]}_{f(t,\boldsymbol{\theta},\boldsymbol{x}_t,\boldsymbol{c})} \tag{3}$$

$$- \mathbb{E}\underbrace{\left[\omega(t)\frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial \boldsymbol{\theta}}\boldsymbol{\epsilon}\right]}_{h_{SDS}(t,\boldsymbol{\theta},\boldsymbol{x},\boldsymbol{c})}, \tag{4}$$

where the second term $\mathbb{E}[h_{SDS}(t,\boldsymbol{\theta},\boldsymbol{x},\boldsymbol{c})] = \mathbf{0}$ simply because it is the expectation of a zero-mean Gaussian

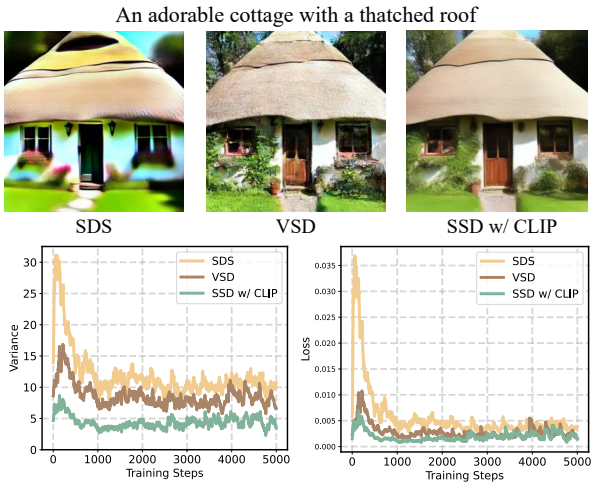

Figure 2: **Comparison between SDS, VSD, and SSD on 2D space.** We monitor the variance of $\boldsymbol{\Delta}_{SDS}$ $\boldsymbol{\Delta}_{VSD}$, and $\boldsymbol{\Delta}_{SSD}$ for every 100 training step. We show that variance level is highly correlated to the final performance and convergence speed.

vector. For VSD, we follow a similar derivation:

$$\boldsymbol{\Delta}_{VSD} = \mathbb{E}\left[\omega(t)\frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial \boldsymbol{\theta}}\sigma_t \nabla \log p_t(\boldsymbol{x}_t|\boldsymbol{y})\right] \tag{5}$$

$$- \underbrace{\mathbb{E}\left[\omega(t)\frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial \boldsymbol{\theta}}\sigma_t \nabla \log q_t(\boldsymbol{x}_t|\boldsymbol{c})\right]}_{h_{VSD}(t,\boldsymbol{\theta},\boldsymbol{x},\boldsymbol{c})}. \tag{6}$$

Taking a closer look, we find that the second term is also zero: $\mathbb{E}[h_{VSD}(t,\boldsymbol{\theta},\boldsymbol{x},\boldsymbol{c})] = \mathbf{0}$. This can be proven by showing that $q_t(\boldsymbol{x}|\boldsymbol{c})$ turns out to be a zero-mean Gaussian distribution or applying the inverse chain rule followed by the fact that the first-order moment of the score function is constantly zero. Moreover, the first term $\mathbb{E}[f(t,\boldsymbol{\theta},\boldsymbol{x},\boldsymbol{c})]$ of both SDS and VSD equals to $-\nabla_{\boldsymbol{\theta}}\,\mathbb{E}_t\left[\mathcal{D}_{\mathrm{KL}}(q_t(\boldsymbol{x}_t|\boldsymbol{c})\|p_t(\boldsymbol{x}_t|\boldsymbol{y}))\right]$. This implies that SDS and VSD are equivalent to gradient descent algorithms, minimizing the distribution discrepancy between the noisy rendered image distribution and the Gaussian perturbed true image distribution. We refer interested readers to Appendix A for the full derivation.

However, in most scenarios, empirical evidence indicates that VSD consistently outperforms SDS, despite both methods aiming to minimize the same objective. To explain this paradox, we posit that the underlying source of their performance disparities is attributed to the *variance* of stochastic simulation of the expected updates by SDS and VSD. The numerical evaluation of Eq. 1 and Eq. 2 typically relies on Monte Carlo estimation over a mini-batch. Unfortunately, rendering a full view from NeRF and performing inference with diffusion models are computationally demanding processes, leading to a constrainted number of rendered views within a single optimization step - often limited to just one, as suggested in previous work (Poole et al., 2022). Additionally, the term related to the score function within the expectation undergoes a denoising procedure, notorious for its instability and high uncertainty, especially when t is large. Hence, despite SDS and VSD having identical means, we argue that the variance of their numerical estimation significantly differs.

We empirically validate this speculation in Fig. 2, wherein we utilize SDS add VSD to sample 2D images from a pre-trained diffusion model, akin to Wang et al. (2023c). We visualize the variance of $\boldsymbol{\Delta}_{SDS}$ and $\boldsymbol{\Delta}_{VSD}$ during the training process. It can be observed that VSD converges faster and yields results with richer details. Moreover, Fig. 2 demonstrates a clear separation between SDS and VSD in terms of the stochastic gradients' variance. Such phenomenon signifies the variance of gradient estimation is highly correlated to the resultant performance.

To gain insight into the variance disparity between SDS and VSD, we connect SDS and VSD via the concept of control variates. As introduced in Sec. 2.2, a control variate is a zero-mean random variable capable of reducing the variance of Monte Carlo estimator when incorporated into the simulated examples. Notably, both $h_{SDS}(t,\boldsymbol{\theta},\boldsymbol{x},\boldsymbol{c})$ and $h_{VSD}(t,\boldsymbol{\theta},\boldsymbol{x},\boldsymbol{c})$ can be regarded as control variates, as confirmed by Eq. 4 and Eq. 6 due to their zero means. Consequently, SDS and VSD can be interpreted as Monte Carlo estimators of the gradient of the KL divergence, integrated with different control variates. As demonstrated in Sec. 2.2, control variate with higher correlation to the estimated variable leads to larger variance reduction. VSD exhibits lower variance primarily because $\nabla \log q_t(\boldsymbol{x}|\boldsymbol{c})$ in control variate h_{VSD} is fine-tuned from $\nabla \log p_t(\boldsymbol{x}|\boldsymbol{c})$, and perhaps resulting in higher correlation compared to the pure Gaussian noises in h_{SDS}.

4 Stein Score Distillation

Having revealed that variance control is one of the key knobs to improve the performance of score distillation, we extend the family of control variates that can be used for score distillation in this section.

4.1 Stein Control Variates for Score Distillation

Our main inspiration is drawn from Oates et al. (2017); Liu (2017); Roeder et al. (2017) that Stein's identity can be served as a powerful and flexible tool to construct zero-mean random variables. We consider Stein's identity associated with any conditional probability

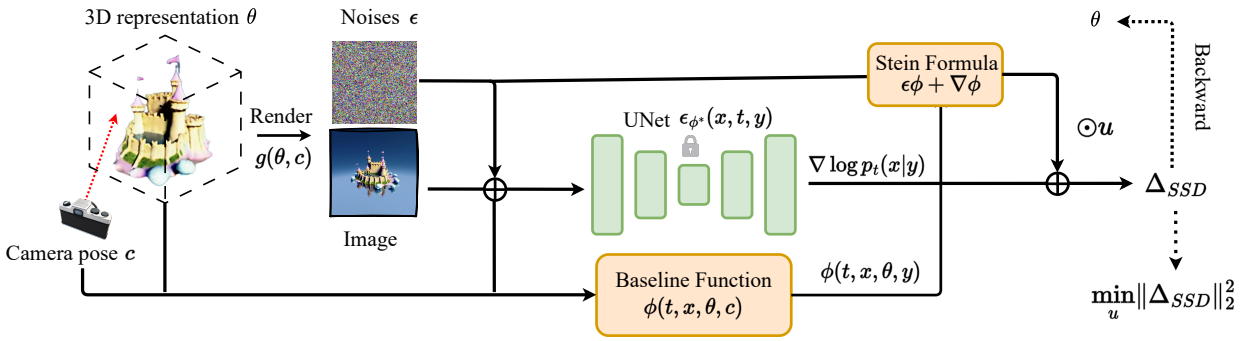

Figure 3: **Pipeline of SteinDreamer.** We incorporate control variates constructed by Stein's identity into a score distillation pipeline, allowing for arbitrary baseline functions. In practice, we implement the baseline functions with a monocular depth or normal estimator.

$p(\boldsymbol{x}_t|\boldsymbol{\theta}, \boldsymbol{c})$ as below:

$$\mathbb{E}_{\boldsymbol{x}_t \sim p(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})} \left[\nabla \log p(\boldsymbol{x}_t|\boldsymbol{\theta}, \boldsymbol{c}) \phi(t, \boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{c}) + \nabla_{\boldsymbol{x}_t} \phi(t, \boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{c})\right] = \mathbf{0}, \tag{7}$$

where $\phi(t, \boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{c})$ is referred to as the *baseline function*, which can be arbitrary scalar-value function satisfying regularity conditions (Stein, 1972; Gorham and Mackey, 2015; Liu et al., 2016). By plugging $q_t(\boldsymbol{x}_t|\boldsymbol{\theta}, \boldsymbol{c})$ into Eq. 7, we can construct our control variate $h_{SSD}(t, \boldsymbol{\theta}, \boldsymbol{c}, \boldsymbol{x}_t)$ as follows:

$$h_{SSD} \triangleq \omega(t) \frac{\partial g(\boldsymbol{\theta}, \boldsymbol{c})}{\partial \boldsymbol{\theta}} \left[\boldsymbol{\epsilon}\phi(t, \boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{c}) + \nabla_{\boldsymbol{x}_t} \phi(t, \boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{c})\right], \tag{8}$$

where $\boldsymbol{x}_t = \alpha_t g(\boldsymbol{\theta}, \boldsymbol{c}) + \sigma_t \boldsymbol{\epsilon}$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Additional details and derivations are provided in Appendix A. The advantage of h_{SSD} lies in its flexibility to define an infinite class of control variates, characterized by arbitrary baseline function $\phi(t, \boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{c})$.

4.2 Variance Minimization via Stein Score Distillation

We propose to adopt h_{SSD} as the control variate for score distillation. In addition to h_{SSD}, we introduce a group of learnable weights $\boldsymbol{\mu} \in \mathbb{R}^D$ to facilitate optimal variance reduction following the standard scheme introduced in Sec. 2.2. Altogether, we present the following update rule, termed as *Stein Score Distillation (SSD)*:

$$\boldsymbol{\Delta}_{SSD} \triangleq \mathbb{E}\left[\omega(t) \frac{\partial g(\boldsymbol{\theta}, \boldsymbol{c})}{\partial \boldsymbol{\theta}} (\sigma_t \nabla \log p_t(\boldsymbol{x}_t|\boldsymbol{y}) + \boldsymbol{\mu} \odot [\boldsymbol{\epsilon}\phi(t, \boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{c}) + \nabla_{\boldsymbol{x}_t} \phi(t, \boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{c})])\right]. \tag{9}$$

Here $\phi(t, \boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{c})$ can be instantiated using any neural network architecture taking 3D parameters, noisy rendered image, and camera pose as the input.

In our experiments, we employ a pre-trained monocular depth estimator, coupled with depth or normal discrepancy penalties to construct $\phi(t, \boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{c})$, as a handy yet effective choice:

$$\phi(t, \boldsymbol{x}_t, \boldsymbol{\theta}, \boldsymbol{c}) \triangleq -\ell(\alpha(\boldsymbol{\theta}, \boldsymbol{c}), \pi_{depth}(\boldsymbol{x}_t)). \tag{10}$$

Here $\pi_{depth}(\cdot)$ can estimate either depth or normal map from noisy observation $\boldsymbol{x}_t$. And $\alpha(\cdot, \cdot)$ is chosen as the corresponding depth or normal renderer of the 3D representation $\boldsymbol{\theta}$, and $\ell(\cdot, \cdot)$ is the Pearson correlation loss when estimating depth map or cosine similarity loss when considering normal map.

As introduced in Sec. 2.2, there exists a closed-form $\boldsymbol{\mu}$ that maximizes the variance reduction. However, it assumes the correlation between the control variate and the random variable of interest is known. Instead, we propose to directly optimize variance by adjusting $\boldsymbol{\mu}$ to minimize the second-order moment of Eq. 9 since its first-order moment is independent of $\boldsymbol{\mu}$:

$$\min_{\boldsymbol{\mu}} \mathbb{E}\left[\left\|\omega(t) \frac{\partial g(\boldsymbol{\theta}, \boldsymbol{c})}{\partial \boldsymbol{\theta}} (\sigma_t \nabla \log p_t(\boldsymbol{x}_t|\boldsymbol{y}) + \boldsymbol{\mu} \odot [\boldsymbol{\epsilon}\phi(t, \boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{c}) + \nabla_{\boldsymbol{x}_t} \phi(t, \boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{c})])\right\|_2^2\right], \tag{11}$$

which essentially imposes a penalty on the gradient norm of $\boldsymbol{\theta}$. We alternate between optimizing $\boldsymbol{\theta}$ and $\boldsymbol{\mu}$ using SSD gradient in Eq. 9 and the objective function in Eq. 11, respectively. We refer to our complete text-to-3D framework as *SteinDreamer*, and its optimization paradigm is illustrated in Fig. 3.

Specifically, during each optimization iteration, SteinDreamer performs the following steps: 1) renders RGB map and depth/normal map from a random view of $\boldsymbol{\theta}$, 2) perturbs the RGB map and obtains the score estimation using a pre-trained diffusion model and monocular depth/normal prediction from a pre-trained depth estimator, 3) computes ϕ via Eq. 10 and its gradient via auto-differentiation to form control variate h_{SSD}, 4) weights the control variate by $\boldsymbol{\mu}$ and combine it with the diffusion score $\nabla \log p_t(\boldsymbol{x}_t|\boldsymbol{y})$, 5) back-propagates

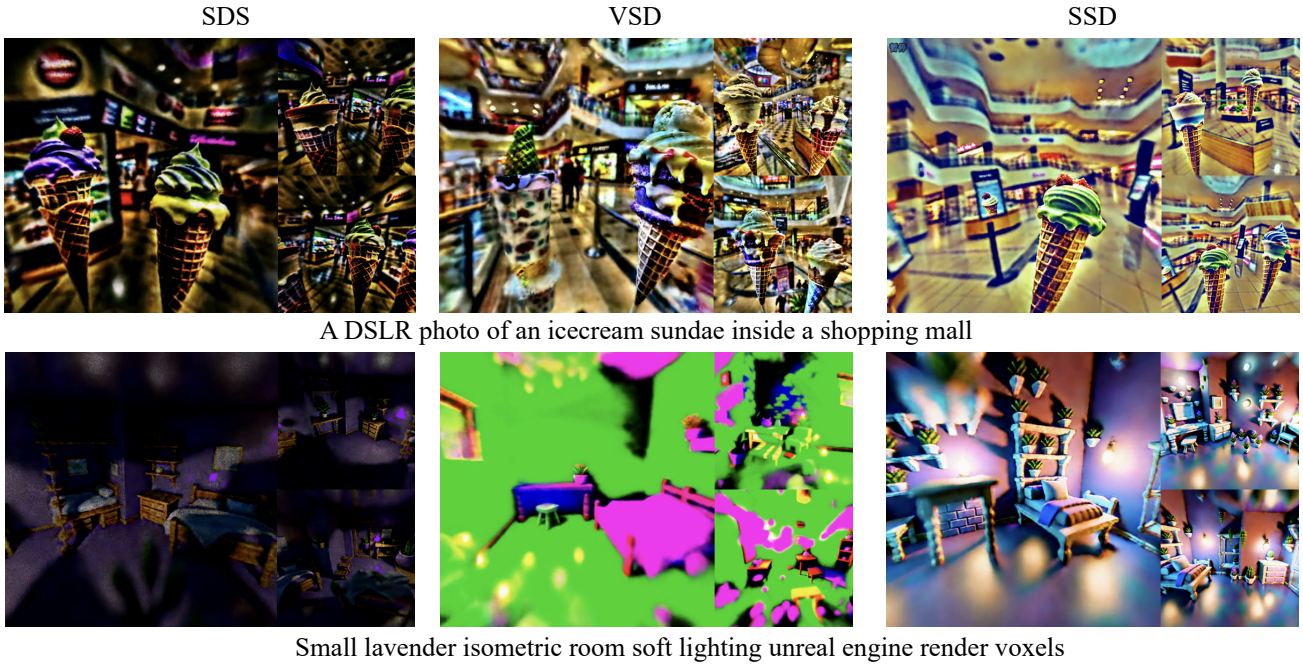

Figure 4: **Scene-level qualitative comparisons.** Compared to existing methods, SteinDreamer w/ normal estimator presents more realistic textures with better details.

$\boldsymbol{\Delta}_{SSD}$ through the chain rule to update 3D parameters $\boldsymbol{\theta}$. In the other fold, SteinDreamer keeps $\boldsymbol{\theta}$ frozen and optimizes $\boldsymbol{\mu}$ to minimize the ℓ_2 norm of the update signals on $\boldsymbol{\theta}$ according to Eq. 11.

4.3 Discussion

SDS is a special case of SSD when taking $\phi(t, \boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{c}) = -1$. Furthermore, the expressive power of SSD surpasses that of VSD, because the baseline function ϕ can directly condition and operate on all relevant variables while $\nabla \log q_t(\boldsymbol{x}_t|\boldsymbol{c})$ in VSD implicitly conditions on $\boldsymbol{\theta}$ through $\boldsymbol{x}_t$ and $\boldsymbol{c}$. This observation suggests the potential for SSD to provide a lower variance in gradient estimation due to its broader range in representing control variates. As demonstrated in Oates et al. (2017), an optimal control variate can be constructed using Stein's identity by carefully selecting ϕ, achieving a zero-variance estimator.

The baseline function ϕ can be also interpreted as a guidance introduced into the distillation process. We contend that control variates when equipped with pre-trained models incorporating appropriate 2D/3D prior knowledge (e.g., depth/normal estimator), are likely to exhibit a higher correlation with the pre-trained score function for image generation. Intuitively, enforcing priors and constraints on the gradient space can also stabilize the training process by regularizing the optimization trajectory.

It is noteworthy that baseline functions other than depth/normal predictors are also applicable. As we discussed above, choosing the right baseline functions can implicitly incorporate desirable prior information. For instance, discriminator (Goodfellow et al., 2014) and CLIP scores (Jain et al., 2022) can be useful to enhance generation quality and text relevance, respectively. We defer more details and examples to Appendix B. Notably, our method supports freely combining arbitrarily many different baseline functions via addition.

To validate our arguments, we present a proof-of-concept experiment on images in Fig. 2, in which we construct a baseline function using CLIP loss (Xu et al., 2023) to regularize image sampling. Please see more details in Appendix D.1. We show that SSD-based stochastic gradient incurs lower variance than the other two baselines, which results in eye-pleasing results and faster convergence.

More discussions and related work can be found in Appendix B and C due to page limit.

5 Experiments

We conduct experiments for both scene-level and object-level text-to-3D generation. The text prompts utilized in object generation are collected from Wang et al. (2023c) while those for scene generation are originally from Höllein et al. (2023) and Zhang et al. (2023). We mainly compare against the seminal works SDS from DreamFusion and VSD from ProlificDreamer. All implementation and hyper-parameters are adopted from

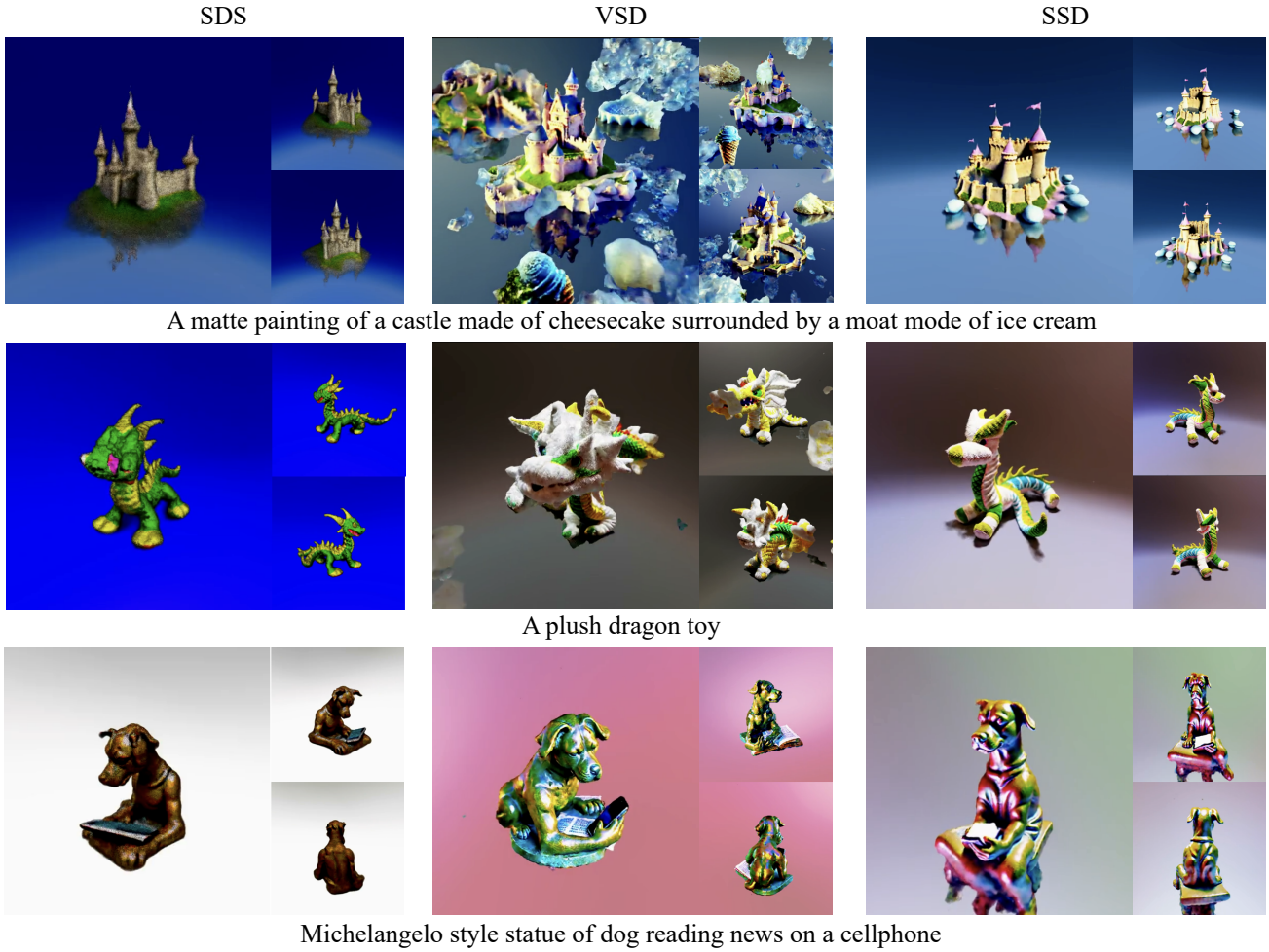

Figure 5: **Object-level qualitative comparisons.** Compared to existing methods, our SteinDreamer w/ normal estimator delivers smoother geometry, more detailed texture, and fewer floater artifacts.

`threestudio` [2] for a fair comparison. In the main text, we mainly present results produced by SteinDreamer with normal estimator while deferring results of SteinDreamer with depth prior to Appendix D.2. Following the common practice, we evaluate VSD with the particle number equal to one.

5.1 Qualitative Evaluation

Large Scene Generation. We evaluate the performance of our method for large-scale scene generation. The detailed comparisons with baselines on 360° scene-level generation is presented in Fig. 4 and Fig. 9. SDS delivers blurry results with unrealistic colors and textures. The results from VSD suffer from the noisy background, and we also observe that the VSD loss can diverge in the texture refining process (Fig. 4). In comparison, we observe that results generated by SteinDreamer are much sharper in appearance and enjoy better details.

Object Centric Generation. We exhibit our object-level qualitative results in Fig. 5 and Fig. 10 for SteinDreamer with normal or depth prior, respectively. Compared with SDS, our SteinDreamer presents novel views with less over-saturation and over-smoothing artifacts. When comparing with VSD, not only does our SteinDreamer generate smoother geometry, but also delivers sharper textures without contamination of floaters. Additionally, it is worth noting that our SteinDreamer can potential alleviate the "Janus" problem by incorporating correct geometric priors, as shown in the dog statue case. We further monitor the variance for all the demonstrated examples during the training stage in Fig. 6. It is clear that our SteinDreamer consistently has lower variance than compared baselines throughout the course of training. This observation supports our theory on the correlation between variance and synthesis quality.

[2] `https://github.com/threestudio-project/threestudio`

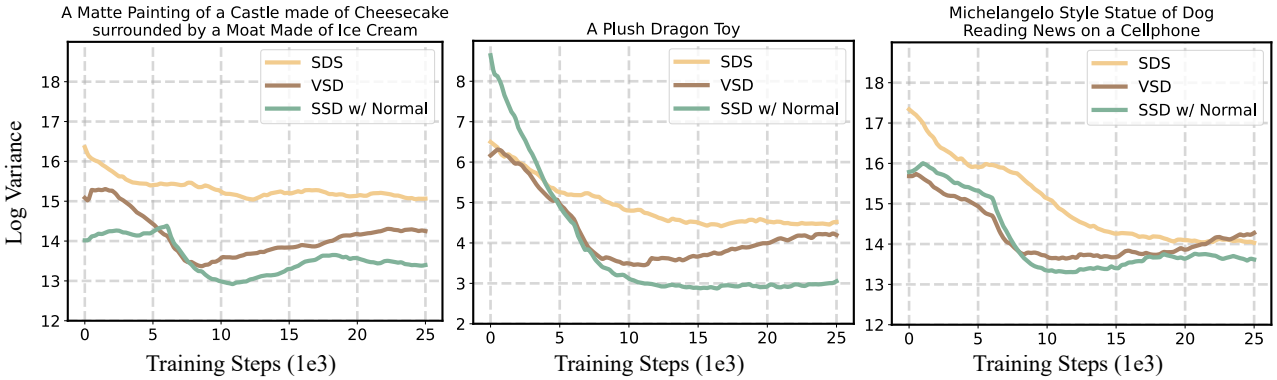

Figure 6: **Variance comparison of Δ_{SDS}, Δ_{VSD}, and Δ_{SSD} with normal estimator.** We visualize how the variance of the investigated three methods for every 1,000 steps. The variance decays as the training converges while Δ_{SSD} consistently achieves lower variance throughout the whole process.

Methods	Scene-level		Object-level	
	CLIP (↓)	FID (↓)	CLIP (↓)	FID (↓)
SDS	0.848±0.068	298.49±57.63	0.898±0.133	282.50±52.11
VSD	0.800±0.051	268.92±49.65	0.763±0.100	271.62±62.77
SSD	0.762±0.039	240.17±45.54	0.720±0.064	251.31±49.70

Table 1: **Quatitative results.** We compare the CLIP ad FID distance (↓ the lower the better) of demonstrated results among different approaches. The best results are marked in color.

Methods	Align. (↑)	Plaus. (↑)	TGC (↑)	Tex. (↑)	Geom. (↑)
SDS	882.72	905.14	846.98	886.47	695.86
VSD	1125.73	1051.98	1018.47	1071.40	1182.80
SSD	1103.94	1104.20	1154.94	1125.25	1324.11

Table 2: **VLM Evaluation.** We evaluate the generated content via VLMs following Wu et al. (2024). All metrics are higher the better (↑). Best numbers are marked in color.

5.2 Quantitative Analysis

Additionally, we report numerical comparison of our methods against other baselines in Tab. 1. We adopt CLIP distance (Xu et al., 2022) and FID (Heusel et al., 2017) as the metrics. For scene generation, we consider 12 text prompts used throughout the whole paper. For object-level generation, we borrow 20 text prompts from Wang et al. (2023c). See Tab. 6 for more details. For each scene or object, we run each algorithm for three times. The reported results is an average of all generated results. We observe that our method consistently outperforms SDS and VSD with a significant gap in both object- and scene-level tasks.

We also quantitatively assess the generated content through vision language models (VLMs) following the framework proposed by Wu et al. (2024). Our evaluation focuses on the following aspects: alignment with the textual prompt (Align.), geometric plausibility (Plaus.), texture-geometry consistency (TGC), texture richness (Tex.), and geometry details (Geom.). The evaluation results are reported in Tab. 2. SSD consistently outperforms in geometric and textural quality, validating the effectiveness of variance reduction in improving generation quality.

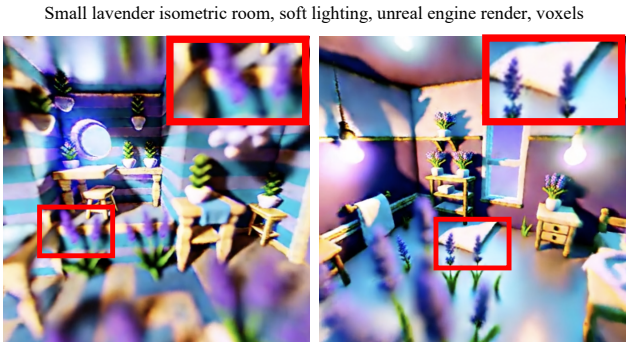

Figure 7: **Ablation study on explicit variance minimization.** We study the effect of turning on/off the optimization step for μ with respect to loss Eq. 11.

5.3 Ablation Studies

Reweighting Coefficients μ. To validate the effectiveness of our proposed components, we conduct ablation studies on whether or not to employ Eq. 11 to minimize second-order moment. The alternative candidate is to fix μ as all-one vector during training. As shown in Fig. 7, when explicitly minimizing the variance, we reach cleaner results with better high-frequency signals. The results when optimizing without variance minimization, on the other hand, turned out to generate blurry geometry and noisy textures. It is worth mentioning that excessive variance reduction may smoothen out some necessary details, especially in the background regions, as the left-hand side result contains more detailed background textures than the right-hand side one.

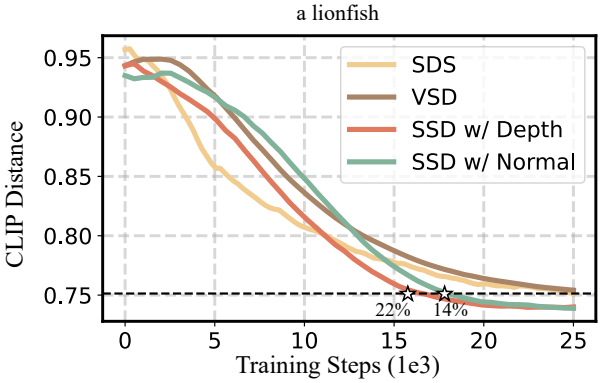

Figure 8: **Convergence speed comparison.** With the help of more stable gradient updates, SteinDreamer accelerates the convergence of the training process by 14%-22% in optimization steps.

Choice of Baseline Functions. Our theoretical framework is capable of incorporating arbitrary prior models into the variance reduction mechanism. There are many potential choices for the baseline functions $\phi(t, \boldsymbol{\theta}, \boldsymbol{x}_t, \boldsymbol{c})$. In particular, we consider constructing control variates via CLIP (Xu et al., 2023) and segmentation models (e.g., Qin et al. (2021)). For the CLIP-based control variate, we define the baseline as the cosine similarity between CLIP embedding of rendered image and user-specified text prompt to enhance their alignment: $\phi = -\langle \text{CLIP}_{image}(g(\boldsymbol{\theta}, \boldsymbol{c})), \text{CLIP}_{text}(\boldsymbol{y}) \rangle$. For the segmentor-based control variate, we compute the cross entropy between the rendered mask $g_{\text{mask}}(\boldsymbol{\theta}, \boldsymbol{c})$ and segmentation map predicted by a pre-trained BASNet (Qin et al., 2021) to penalize disconnected geometries: $\phi = -\text{CE}(g_{\text{mask}}(\boldsymbol{\theta}, \boldsymbol{c}), \text{BASNet}(\boldsymbol{x}_t))$. We will elaborate on details and more baseline functions in Appendix B. We empirically compare our current choice based on the depth estimator against CLIP-based and segmentor-based control variates. The results are summarized in Tab. 3. Compared with the Gaussian noise control variate h_{SDS}, we find that our Stein-based control variate generally improves generation quality, with flexibility in the choice of baseline functions. Moreover, selecting a baseline function that incorporates more 3D prior knowledge (e.g., depth estimator) appears to be particularly effective than pre-trained 2D models.

Baselines	Gaussian	Depth Pred.	CLIP	Segmenter
CLIP (↓)	0.848	0.762	0.745	0.818
FID (↓)	298.49	240.17	255.51	260.64

Table 3: **Different choices of baseline functions.** The performance is evaluated based on CLIP score and FID. The "Gaussian" refers to the Gaussian noise control variate adopted by SDS (see Eq. 4).

5.4 Convergence Speed

We also study the convergence speed of our methods as well as compared baselines. Specifically, we use the average CLIP distance (Xu et al., 2022) between the rendered images and the input text prompts as the quality metric. During the training process, we render the 3D assets into multi-view images every 1,000 training steps. In each training step, the diffusion model is inference twice through the classifier-free guidance, which is the same protocol in all compared methods. In Fig. 8, we profile the training steps needed for each approach to reach 0.75 CLIP distance as a desirable threshold. We observe that the proposed SteinDreamer can effectively attain rapid and superior convergence, saving 14%-22% calls of diffusion models. Our SteinDreamer utilizes fewer number of score function evaluations to achieve distilled 3D results that are more aligned with the text prompts. This can be explained by the classic optimization theory that lower variance in our distillation process can speed up convergence. Moreover, since SteinDreamer avoids inferencing and fine-tuning another diffusion model, each iteration of SSD is approximately 30% faster than VSD. We provide a detailed analysis in Appendix D.4. In Appendix D.5, we demonstrate that a modest increase in training steps for SDS and VSD fails to overcome the issues of high-variance gradients to match SSD performance.

6 Conclusion

In this work, we present SteinDreamer, revealing a more general solution to reduce variance for score distillation. Our Stein Score Distillation (SSD) incorporates control variates through Stein identity, admitting arbitrary baseline functions conditioned on all relevant variables with any guidance priors. The experimental results suggest that SSD can effectively reduce the distillation variance and consistently improve visual quality for both object- and scene-level generations. We also showcase that SSD achieves faster and better convergence than existing methods with the help of more stable gradient updates.

Acknowledgements

Work was done during P Wang's internship at Meta Reality Labs. P Wang sincerely appreciates insightful discussions with Zhaoyang Lv, Xiaoyu Xiang, Amit Kumar, Jinhui Xiong, and Varun Nagaraja. P Wang also thanks Ruisi Cai for helping plot figures. Any statements, opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of their employers or the supporting entities.

References

Armandpour, M., Zheng, H., Sadeghian, A., Sadeghian, A., and Zhou, M. (2023). Re-imagine the negative prompt algorithm: Transform 2d diffusion into 3d, alleviate janus problem and beyond. *arXiv preprint arXiv:2304.04968*.

Cai, R., Yang, G., Averbuch-Elor, H., Hao, Z., Belongie, S., Snavely, N., and Hariharan, B. (2020). Learning gradient fields for shape generation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 364–381. Springer.

Chan, E. R., Lin, C. Z., Chan, M. A., Nagano, K., Pan, B., De Mello, S., Gallo, O., Guibas, L. J., Tremblay, J., Khamis, S., et al. (2022). Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16123–16133.

Chen, J., Zhu, J., and Song, L. (2017). Stochastic training of graph convolutional networks with variance reduction. *arXiv preprint arXiv:1710.10568*.

Chen, L. H. (1975). Poisson approximation for dependent trials. *The Annals of Probability*, 3(3):534–545.

Chen, R., Chen, Y., Jiao, N., and Jia, K. (2023a). Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. *arXiv preprint arXiv:2303.13873*.

Chen, Y., Zhang, C., Yang, X., Cai, Z., Yu, G., Yang, L., and Lin, G. (2023b). It3d: Improved text-to-3d generation with explicit view synthesis. *arXiv preprint arXiv:2308.11473*.

Davies, C. T., Follana, E., Gray, A., Lepage, G., Mason, Q., Nobes, M., Shigemitsu, J., Trottier, H., Wingate, M., Aubin, C., et al. (2004). High-precision lattice qcd confronts experiment. *Physical Review Letters*, 92(2):022001.

Detommaso, G., Cui, T., Marzouk, Y., Spantini, A., and Scheichl, R. (2018). A stein variational newton method. *Advances in Neural Information Processing Systems*, 31.

Dhariwal, P. and Nichol, A. (2021). Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794.

Garrigos, G. and Gower, R. M. (2023). Handbook of convergence theorems for (stochastic) gradient methods. *arXiv preprint arXiv:2301.11235*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, 27.

Gorham, J. and Mackey, L. (2015). Measuring sample quality with stein's method. *Advances in neural information processing systems*, 28.

Hertz, A., Aberman, K., and Cohen-Or, D. (2023). Delta denoising score. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2328–2337.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30.

Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851.

Höllein, L., Cao, A., Owens, A., Johnson, J., and Nießner, M. (2023). Text2room: Extracting textured 3d meshes from 2d text-to-image models. *arXiv preprint arXiv:2303.11989*.

Hong, S., Ahn, D., and Kim, S. (2023). Debiasing scores and prompts of 2d diffusion for robust text-to-3d generation. *arXiv preprint arXiv:2303.15413*.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2021). Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Huang, Y., Wang, J., Shi, Y., Qi, X., Zha, Z.-J., and Zhang, L. (2023). Dreamtime: An improved optimization strategy for text-to-3d content creation. *arXiv preprint arXiv:2306.12422*.

Jain, A., Mildenhall, B., Barron, J. T., Abbeel, P., and Poole, B. (2022). 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.

Jun, H. and Nichol, A. (2023). Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*.

Kajiya, J. T. (1986). The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 143–150.

Katzir, O., Patashnik, O., Cohen-Or, D., and Lischinski, D. (2023). Noise-free score distillation. *arXiv preprint arXiv:2310.17590*.

Kim, S., Lee, K., Choi, J. S., Jeong, J., Sohn, K., and Shin, J. (2023). Collaborative score distillation for consistent visual editing. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Lin, C.-H., Gao, J., Tang, L., Takikawa, T., Zeng, X., Huang, X., Kreis, K., Fidler, S., Liu, M.-Y., and Lin, T.-Y. (2023). Magic3d: High-resolution

text-to-3d content creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 300–309.

Liu, H., Feng, Y., Mao, Y., Zhou, D., Peng, J., and Liu, Q. (2017). Action-depedent control variates for policy optimization via stein's identity. *arXiv preprint arXiv:1710.11198*.

Liu, Q. (2017). Stein variational gradient descent as gradient flow. *Advances in neural information processing systems*, 30.

Liu, Q., Lee, J., and Jordan, M. (2016). A kernelized stein discrepancy for goodness-of-fit tests. In *International conference on machine learning*, pages 276–284. PMLR.

Liu, Q. and Wang, D. (2016). Stein variational gradient descent: A general purpose bayesian inference algorithm. *Advances in neural information processing systems*, 29.

Liu, R., Wu, R., Van Hoorick, B., Tokmakov, P., Zakharov, S., and Vondrick, C. (2023). Zero-1-to-3: Zero-shot one image to 3d object. *arXiv preprint arXiv:2303.11328*.

Luo, W., Hu, T., Zhang, S., Sun, J., Li, Z., and Zhang, Z. (2024). Diff-instruct: A universal approach for transferring knowledge from pre-trained diffusion models. *Advances in Neural Information Processing Systems*, 36.

Mardani, M., Song, J., Kautz, J., and Vahdat, A. (2023). A variational perspective on solving inverse problems with diffusion models. *arXiv preprint arXiv:2305.04391*.

Max, N. (1995). Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108.

McAllister, D., Ge, S., Huang, J.-B., Jacobs, D. W., Efros, A. A., Holynski, A., and Kanazawa, A. (2024). Rethinking score distillation as a bridge between image distributions. *arXiv preprint arXiv:2406.09417*.

Metzer, G., Richardson, E., Patashnik, O., Giryes, R., and Cohen-Or, D. (2023). 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.

Meyn, S. (2008). *Control techniques for complex networks*. Cambridge University Press.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. (2020). Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer.

Müller, T., Evans, A., Schied, C., and Keller, A. (2022). Instant neural graphics primitives with a multiresolution hash encoding. *arXiv preprint arXiv:2201.05989*.

Müller, T., Rousselle, F., Keller, A., and Novák, J. (2020). Neural control variates. *ACM Transactions on Graphics (TOG)*, 39(6):1–19.

Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., and Chen, M. (2021). Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*.

Nichol, A., Jun, H., Dhariwal, P., Mishkin, P., and Chen, M. (2022). Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*.

Oates, C. J., Girolami, M., and Chopin, N. (2017). Control functionals for monte carlo integration. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 79(3):695–718.

Poole, B., Jain, A., Barron, J. T., and Mildenhall, B. (2022). Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*.

Qin, X., Fan, D.-P., Huang, C., Diagne, C., Zhang, Z., Sant'Anna, A. C., Suarez, A., Jagersand, M., and Shao, L. (2021). Boundary-aware segmentation network for mobile and web applications. *arXiv preprint arXiv:2101.04704*.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. (2022). Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. (2021). Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR.

Roeder, G., Wu, Y., and Duvenaud, D. K. (2017). Sticking the landing: Simple, lower-variance gradient estimators for variational inference. *Advances in Neural Information Processing Systems*, 30.

Sauer, A., Lorenz, D., Blattmann, A., and Rombach, R. (2023). Adversarial diffusion distillation. *arXiv preprint arXiv:2311.17042*.

Seo, J., Jang, W., Kwak, M.-S., Ko, J., Kim, H., Kim, J., Kim, J.-H., Lee, J., and Kim, S. (2023). Let 2d diffusion model know 3d-consistency for robust text-to-3d generation. *arXiv preprint arXiv:2303.07937*.

Shao, R., Sun, J., Peng, C., Zheng, Z., Zhou, B., Zhang, H., and Liu, Y. (2023). Control4d: Dynamic portrait editing by learning 4d gan from 2d diffusion-based editor. *arXiv preprint arXiv:2305.20082*.

Shen, T., Gao, J., Yin, K., Liu, M.-Y., and Fidler, S. (2021). Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. *Advances in Neural Information Processing Systems*, 34:6087–6101.

Shue, J. R., Chan, E. R., Po, R., Ankner, Z., Wu, J., and Wetzstein, G. (2022). 3d neural field generation using triplane diffusion. *arXiv preprint arXiv:2211.16677*.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR.

Song, Y. and Ermon, S. (2019). Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. (2020). Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*.

Stein, C. (1972). A bound for the error in the normal approximation to the distribution of a sum of dependent random variables. In *Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability, Volume 2: Probability Theory*, volume 6, pages 583–603. University of California Press.

Sutton, R. S., Barto, A. G., et al. (1998). *Introduction to reinforcement learning*, volume 135. MIT press Cambridge.

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. (1999). Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12.

Tsalicoglou, C., Manhardt, F., Tonioni, A., Niemeyer, M., and Tombari, F. (2023). Textmesh: Generation of realistic 3d meshes from text prompts. *arXiv preprint arXiv:2304.12439*.

Wang, H., Du, X., Li, J., Yeh, R. A., and Shakhnarovich, G. (2023a). 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.

Wang, P., Xu, D., Fan, Z., Wang, D., Mohan, S., Iandola, F., Ranjan, R., Li, Y., Liu, Q., Wang, Z., et al. (2023b). Taming mode collapse in score distillation for text-to-3d generation. *arXiv preprint arXiv:2401.00909*.

Wang, Z., Lu, C., Wang, Y., Bao, F., Li, C., Su, H., and Zhu, J. (2023c). Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *arXiv preprint arXiv:2305.16213*.

Warburg, F., Weber, E., Tancik, M., Holynski, A., and Kanazawa, A. (2023). Nerfbusters: Removing ghostly artifacts from casually captured nerfs. *arXiv preprint arXiv:2304.10532*.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256.

Wu, J., Zhang, C., Xue, T., Freeman, B., and Tenenbaum, J. (2016). Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *Advances in neural information processing systems*, 29.

Wu, T., Yang, G., Li, Z., Zhang, K., Liu, Z., Guibas, L., Lin, D., and Wetzstein, G. (2024). Gpt-4v (ision) is a human-aligned evaluator for text-to-3d generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22227–22238.

Xu, D., Jiang, Y., Wang, P., Fan, Z., Wang, Y., and Wang, Z. (2022). Neurallift-360: Lifting an in-the-wild 2d photo to a 3d object with 360 {\deg} views. *arXiv preprint arXiv:2211.16431*.

Xu, H., Xie, S., Tan, X. E., Huang, P.-Y., Howes, R., Sharma, V., Li, S.-W., Ghosh, G., Zettlemoyer, L., and Feichtenhofer, C. (2023). Demystifying clip data. *arXiv preprint arXiv:2309.16671*.

Yang, G., Huang, X., Hao, Z., Liu, M.-Y., Belongie, S., and Hariharan, B. (2019). Pointflow: 3d point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4541–4550.

Yin, T., Gharbi, M., Zhang, R., Shechtman, E., Durand, F., Freeman, W. T., and Park, T. (2024). One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6613–6623.

Yu, X., Guo, Y.-C., Li, Y., Liang, D., Zhang, S.-H., and Qi, X. (2023). Text-to-3d with classifier score distillation. *arXiv preprint arXiv:2310.19415*.

Zhang, Q., Wang, C., Siarohin, A., Zhuang, P., Xu, Y., Yang, C., Lin, D., Zhou, B., Tulyakov, S., and Lee, H.-Y. (2023). Scenewiz3d: Towards text-guided 3d scene composition. *arXiv preprint arXiv:2312.08885*.

Zhou, M., Zheng, H., Wang, Z., Yin, M., and Huang, H. (2024). Score identity distillation: Exponentially fast distillation of pretrained diffusion models for one-step generation. In *Forty-first International Conference on Machine Learning*.

Zhu, J. and Zhuang, P. (2023). Hifa: High-fidelity text-to-3d with advanced diffusion guidance. *arXiv preprint arXiv:2305.18766*.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Not Applicable]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [No]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [No]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [No]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A Deferred Derivations

Gradient of KL divergence. Let $\boldsymbol{\theta}$ parameterize the underlying 3D representation, such as NeRF (Mildenhall et al., 2020). We intend to optimize $\boldsymbol{\theta}$ such that each view matches the prior of 2D distribution. This can be formulated by minimizing the KL divergence below [3]:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{t,\boldsymbol{c}\sim p(\boldsymbol{c})}\, \mathcal{D}_{\mathrm{KL}}(q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})\|p_t(\boldsymbol{x}_t|\boldsymbol{y})), \tag{12}$$

where $\boldsymbol{c}$ is the camera pose sampled from a prior distribution, $\boldsymbol{y}$ is the user-specified text prompt, and $q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c}) = \mathcal{N}(\boldsymbol{x}_t|\alpha_t g(\boldsymbol{\theta},\boldsymbol{c}), \sigma_t^2\boldsymbol{I})$, where $g(\boldsymbol{\theta},\boldsymbol{c})$ is a differentiable renderer that displays scene $\boldsymbol{\theta}$ from the camera angle $\boldsymbol{c}$.

To optimize Eq. 12, we take the gradient in terms of $\boldsymbol{\theta}$ and derive the following update formula:

$$\nabla_{\boldsymbol{\theta}}\, \mathbb{E}_{t,\boldsymbol{c}}\, \mathcal{D}_{\mathrm{KL}}(q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})\|p_t(\boldsymbol{x}_t|\boldsymbol{y})) = \mathbb{E}_{t,\boldsymbol{c}} \nabla_\theta\, \mathcal{D}_{\mathrm{KL}}(q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})\|p_t(\boldsymbol{x}_t|\boldsymbol{y})) \tag{13}$$

$$= \mathbb{E}_{t,\boldsymbol{c}}\, \nabla_{\boldsymbol{\theta}}\, \mathbb{E}_{\boldsymbol{x}_t\sim q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})} \left[\log \frac{q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})}{p_t(\boldsymbol{x}_t|\boldsymbol{y})}\right] \tag{14}$$

$$= \mathbb{E}_{t,\boldsymbol{c},\boldsymbol{\epsilon}\sim\mathcal{N}(\boldsymbol{0},\sigma_t^2\boldsymbol{I})} \left[\underbrace{\nabla_{\boldsymbol{\theta}} \log q_t(\alpha_t g(\boldsymbol{\theta},\boldsymbol{c}) + \boldsymbol{\epsilon}|\boldsymbol{\theta},\boldsymbol{c})}_{(a)} - \underbrace{\nabla_{\boldsymbol{\theta}} \log p_t(\alpha_t g(\boldsymbol{\theta},\boldsymbol{c}) + \boldsymbol{\epsilon}|\boldsymbol{y})}_{(b)}\right] \tag{15}$$

We notice that $q_t(\alpha_t g(\boldsymbol{\theta},\boldsymbol{c}) + \boldsymbol{\epsilon}|\boldsymbol{\theta},\boldsymbol{c}) = \mathcal{N}(\boldsymbol{\epsilon}|\boldsymbol{0},\sigma_t^2\boldsymbol{I})$, which is independent of $\boldsymbol{\theta}$. Thus $(a) = \boldsymbol{0}$. For term (b), we have:

$$\nabla_{\boldsymbol{\theta}} \log p_t(\alpha_t g(\boldsymbol{\theta},\boldsymbol{c}) + \boldsymbol{\epsilon}|\boldsymbol{y}) = \alpha_t \frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial \boldsymbol{\theta}} \nabla \log p_t(\alpha_t g(\boldsymbol{\theta},\boldsymbol{c}) + \boldsymbol{\epsilon}|\boldsymbol{y}). \tag{16}$$

Therefore, $\boldsymbol{\theta}$ should be iteratively updated by:

$$\mathbb{E}_{t,\boldsymbol{c},\boldsymbol{\epsilon}}\left[\alpha_t \frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial \boldsymbol{\theta}} \nabla \log p_t(\alpha_t g(\boldsymbol{\theta},\boldsymbol{c}) + \boldsymbol{\epsilon}|\boldsymbol{y})\right] = \mathbb{E}_{t,\boldsymbol{c},\boldsymbol{x}_t\sim q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})}\left[\alpha_t \frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial \boldsymbol{\theta}} \nabla \log p_t(\boldsymbol{x}_t|\boldsymbol{y})\right] \tag{17}$$

SDS equals to the gradient of KL. By the following derivation, we demonstrate that SDS essentially minimizes the KL divergence: $\boldsymbol{\Delta}_{SDS} = \nabla_{\boldsymbol{\theta}}\, \mathbb{E}_{t,\boldsymbol{c}}\, \mathcal{D}_{\mathrm{KL}}(q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})\|p_t(\boldsymbol{x}_t|\boldsymbol{y}))$:

$$\mathbb{E}_{t,\boldsymbol{c},\boldsymbol{x}_t\sim q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})}\left[\frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial \boldsymbol{\theta}} \left(\nabla \log p_t(\boldsymbol{x}_t|\boldsymbol{y}) - \boldsymbol{\epsilon}\right)\right] \tag{18}$$

$$= \mathbb{E}_{t,\boldsymbol{c},\boldsymbol{x}_t\sim q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})}\left[\alpha_t \frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial \boldsymbol{\theta}} \nabla \log p_t(\boldsymbol{x}_t|\boldsymbol{y})\right] - \underbrace{\mathbb{E}_{t,\boldsymbol{c},\boldsymbol{\epsilon}\sim\mathcal{N}(\boldsymbol{0},\sigma_t^2\boldsymbol{I})}\left[\alpha_t \frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial \boldsymbol{\theta}} \boldsymbol{\epsilon}\right]}_{=\boldsymbol{0}}. \tag{19}$$

VSD equals to the gradient of KL. We show that VSD also equals to the gradient of KL $\boldsymbol{\Delta}_{VSD} = \nabla_{\boldsymbol{\theta}}\, \mathbb{E}_{t,\boldsymbol{c}}\, \mathcal{D}_{\mathrm{KL}}(q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})\|p_t(\boldsymbol{x}_t|\boldsymbol{y}))$ due to the simple fact that the first-order of score equals to zero:

$$\mathbb{E}_{t,\boldsymbol{c},\boldsymbol{x}_t\sim q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})}\left[\alpha_t \frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial \boldsymbol{\theta}} \nabla \log q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})\right] = \mathbb{E}_{t,\boldsymbol{c},\boldsymbol{x}_t\sim q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})}\left[\nabla_{\boldsymbol{\theta}} \log q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})\right] \tag{20}$$

$$= \mathbb{E}_{t,\boldsymbol{c}}\left[\int \frac{\nabla_{\boldsymbol{\theta}} q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})}{q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})} q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c}) d\boldsymbol{x}_t\right] \tag{21}$$

$$= \mathbb{E}_{t,\boldsymbol{c}}\left[\nabla_{\boldsymbol{\theta}} \int q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c}) d\boldsymbol{x}_t\right] = \boldsymbol{0}. \tag{22}$$

Control Variate for SSD. Due to Stein's identity, the following is constantly zero:

$$\mathbb{E}_{\boldsymbol{x}_t\sim q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})}\left[\nabla \log q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})\phi(t,\boldsymbol{\theta},\boldsymbol{x}_t,\boldsymbol{c}) + \nabla_{\boldsymbol{x}_t}\phi(t,\boldsymbol{\theta},\boldsymbol{x}_t,\boldsymbol{c})\right] = \boldsymbol{0}. \tag{23}$$

[3] Without loss of generality, we intend to omit coefficients $\omega(t)$ in all derivations for the sake of simplicity.

Plug into Eq. 17, we can obtain:

$$\mathbb{E}_{t,\boldsymbol{c},\boldsymbol{x}_t\sim q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})}\left[\alpha_t\frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial\boldsymbol{\theta}}\nabla\log p_t(\boldsymbol{x}_t|\boldsymbol{y})\right] \tag{24}$$

$$=\mathbb{E}_{t,\boldsymbol{c}}\left[\frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial\boldsymbol{\theta}}\,\mathbb{E}_{\boldsymbol{x}_t\sim q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})}\left[\nabla\log p_t(\boldsymbol{x}_t|\boldsymbol{y})+\nabla\log q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})\phi(t,\boldsymbol{\theta},\boldsymbol{x}_t,\boldsymbol{c})+\nabla_{\boldsymbol{x}_t}\phi(t,\boldsymbol{\theta},\boldsymbol{x}_t,\boldsymbol{c})\right]\right] \tag{25}$$

$$=\mathbb{E}_{t,\boldsymbol{c}}\left[\frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial\boldsymbol{\theta}}\,\mathbb{E}_{\boldsymbol{x}_t\sim q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})}\left[\nabla\log p_t(\boldsymbol{x}_t|\boldsymbol{y})+\boldsymbol{\epsilon}\phi(t,\boldsymbol{\theta},\boldsymbol{x}_t,\boldsymbol{c})+\nabla_{\boldsymbol{x}_t}\phi(t,\boldsymbol{\theta},\boldsymbol{x}_t,\boldsymbol{c})\right]\right] \tag{26}$$

$$=\mathbb{E}_{t,\boldsymbol{c},\boldsymbol{\epsilon}}\left[\frac{\partial g(\boldsymbol{\theta},\boldsymbol{c})}{\partial\boldsymbol{\theta}}(\nabla\log p_t(\boldsymbol{x}_t|\boldsymbol{y})+\boldsymbol{\epsilon}\phi(t,\boldsymbol{\theta},\boldsymbol{x}_t,\boldsymbol{c})+\nabla_{\boldsymbol{x}_t}\phi(t,\boldsymbol{\theta},\boldsymbol{x}_t,\boldsymbol{c}))\right], \tag{27}$$

where Eq. 26 can be derived by noticing $q_t(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{c})$ follows from a Gaussian distribution.

B Deferred Discussion

In this section, we continue our discussion from Sec. 4.3.

How does baseline function reduce variance? The baseline function ϕ can be regarded as a guidance introduced into the distillation process. We contend that control variates when equipped with pre-trained models incorporating appropriate 2D/3D prior knowledge, are likely to exhibit a higher correlation with the score function. Intuitively, enforcing priors and constraints on the gradient space can also stabilize the training process by regularizing the optimization trajectory. Therefore, in our empirical design, the inclusion of geometric information expressed by a pre-trained depth estimator is expected to result in superior variance reduction compared to SSD and VSD.

Detailed comparison with VSD. In VSD, the adopted control variate $\nabla\log q_t(\boldsymbol{x}_t|\boldsymbol{c})$ is fine-tuned based on a pre-trained score function using LoRA (Hu et al., 2021). However, this approach presents two primary drawbacks: 1) The trained control variate may not fully converge to the desired score function, potentially resulting in non-zero mean and biased gradient estimations. 2) Fine-tuning another large diffusion model also significantly increases the computation expenses. Our SSD effectively circumvents these two limitations. Firstly, the control variate in SSD is provably zero-mean, as per Stein's identity. Additionally, the computational cost associated with differentiating the frozen ϕ and optimizing the weights $\boldsymbol{u}$ remains manageable. We verify the computational efficiency of SSD in Appendix D.4.

Other baseline functions. Baseline functions other than depth/normal predictors are also applicable. Here we provide some tentative options for future exploration. A foreground-background segmenter coupled with a classification loss can be useful to mitigate the artifacts of missing parts in generated 3D objects. A discriminator from a pre-trained GAN (Goodfellow et al., 2014) associated with the discriminative loss can be utilized to implement the baseline function to improve the fidelity of each view. Similarly, CLIP loss (Jain et al., 2022) inducted baseline function might help increase relevance with specified text. Multi-view prior such as Zero123 (Liu et al., 2023) can be also introduced by sampling another view as a function of $\boldsymbol{\theta}$ and comparing it with the current view $\boldsymbol{x}_t$. Our method can also freely combine all these aforementioned baseline functions.

C Other Related Work

Prior techniques to improve score distillation. There have been various methods proposed to enhance the effectiveness of score distillation. SJC (Wang et al., 2023a) derives the score Jacobian chaining method through a "Perturb and Average Scoring" argument, concurrent to SDS (Poole et al., 2022). Approaches like Magic3D (Lin et al., 2023) and Fantasia3D (Chen et al., 2023a) leverage mesh and DMTet (Shen et al., 2021) to separate the optimization processes for geometry and texture. Techniques such as TextMesh (Tsalicoglou et al., 2023) and 3DFuse (Seo et al., 2023) integrate depth-conditioned text-to-image diffusion priors, facilitating geometry-aware texturing. Other works, such as Score Debiasing (Hong et al., 2023) and Perp-Neg (Armandpour et al., 2023), focus on refining text prompts for improved 3D generation. Meanwhile, DreamTime (Huang et al., 2023) and RED-Diff (Mardani et al., 2023) explore optimizing timestep scheduling during the distillation process.

HIFA (Zhu and Zhuang, 2023) introduces multiple diffusion steps to trace for more accurate score estimation for distillation. Additionally, score distillation can be combined with auxiliary losses, such as CLIP loss (Xu et al., 2022) and adversarial loss (Shao et al., 2023; Chen et al., 2023b), to further improve results.

Stein methods in score distillation. The work most similar to ours is, perhaps, Collaborative Score Distillation (CSD) (Kim et al., 2023), which samples latent parameters via Stein Variational Gradient Descent (SVGD). While both methods are grounded in Stein's method, the underlying principles significantly differ. In CSD, the SVGD-based update takes the form of the Stein discrepancy: $\max_{\phi\in\mathcal{F}} \mathbb{E}_{\boldsymbol{x}\sim q(\boldsymbol{x})}[\phi(\boldsymbol{x})\nabla \log p(\boldsymbol{x}) + \nabla_{\boldsymbol{x}}\phi(\boldsymbol{x})]$, where $\phi(\boldsymbol{x})$ is often interpreted as an update direction constrained by a function class $\mathcal{F}$ (RBF kernel space in Kim et al. (2023)). In contrast, our update rule appends a zero-mean random variable via the Stein identity after the raw gradient of the KL divergence (Eq. 9), where $\phi(\boldsymbol{x})$ typically represents a pre-defined baseline function. The potential rationale behind CSD to reducing variance lies in introducing the RBF kernel as a prior to constrain the solution space by modeling pairwise relations between data samples. Our SSD is centered around constructing a more general control variate that correlates with the random variable of interest, featuring zero mean but variance reduction.

Concurrent progress in score distillation. While we were preparing this manuscript, several new score distillation techniques for text-to-3D generation emerged. Noise-Free Score Distillation (NFSD) (Katzir et al., 2023) and Classifier Score Distillation (Yu et al., 2023) propose to drop the random noises in Eq. 1 while introducing a score with negative prompts and reweighting each score components to balance the effects of unconditional, conditional, and contrastive generative priors (McAllister et al., 2024). Entropic Score Distillation (ESD) (Wang et al., 2023b) shows that introducing CFG trick to the score of rendered image distribution in Eq. 2 can recover variational objectives to match the pre-trained image and rendered image distribution. In addition to 3D generation, score distillation has also been used for image editing (Hertz et al., 2023) and distilling multi-step denoisers to one-step generators (Sauer et al., 2023; Luo et al., 2024; Zhou et al., 2024; Yin et al., 2024).

D Additional Experiments

D.1 Implementation Details

Image Sampling. In our 2D experiment in Fig. 2, one could simply regard $g(\boldsymbol{\theta}) = \boldsymbol{\theta}$ where $\boldsymbol{\theta}$ is an image representation. We train all algorithms for 5k steps with learning rate 1e-2. For VSD, we update its LoRA with learning rate 1e-4. When sampling with SSD, we utilize CLIP guidance to construct the baseline function, akin to Xu et al. (2022). In particular, baseline function is defined as $\phi(t, \boldsymbol{x}_t, \boldsymbol{\theta}, \boldsymbol{y}) = -\langle \text{CLIP}_{image}(\boldsymbol{\theta}), \text{CLIP}_{text}(\boldsymbol{y})\rangle$.

3D Generation. All of baselines are implemented based on the `threestudio` framework. For fairness, we only compare the results yielded in the coarse generation stage for object generation without geometry refinement specified in ProlificDreamer. We employ hash-encoded NeRF (Müller et al., 2022) as the underlying 3D representation, and disentangle representations for foreground and background separately. All scenes are trained for 25k steps with a learning rate of 1e-2. At each iteration, we randomly sample one view for supervision. We progressively increase rendering resolution from 64×64 resolution to 256×256 resolution after 5k steps. View-dependent prompting is enabled to alleviate Janus problem. Other hyperparameters are kept consistent with the default values. In our implementation of SteinDreamer, the depth estimator is operated on images decoded from the latent space. We further scale the baseline function by a coefficient 1e-2. The pre-trained depth estimator is based on a hybrid architecture of transformer and ResNet. We convert the estimated inverse depth to normal by directly taking spatial derivatives and normalization. Such operation is equivalent to the standard computation via normalizing spatial derivatives of non-inverse depth. The rendered normal is analytically computed as the gradient of the density field. Additionally, we reweight the pre-average loss map via the alpha mask rendered from NeRF. The weighting coefficients $\boldsymbol{\mu}$ are initialized with an all-one vector.

D.2 More Qualitative Results

We demonstrate deferred results' of SteinDreamer with depth estimator in Fig. 10 and Fig. 5. Consistent with our observation in Sec. 5, our method yields smooth and consistent renderings. We refer interested readers to our supplementary materials for video demos. Moreover, we visualize the comparison on variance during object

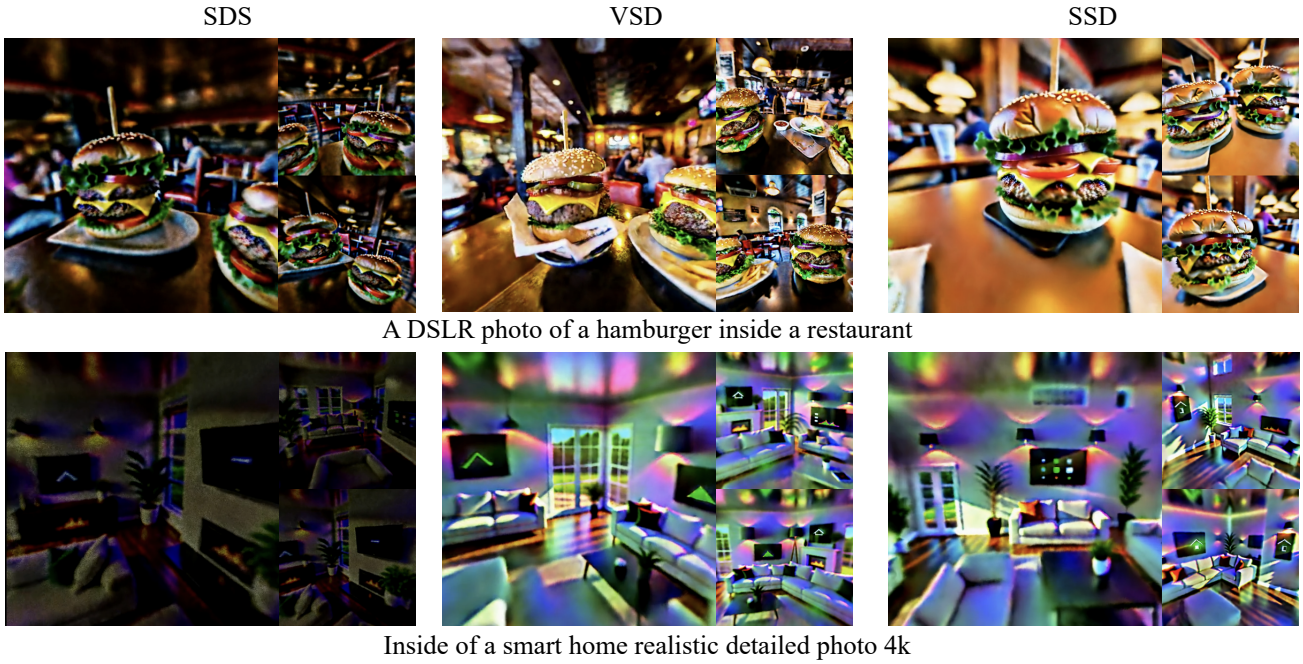

Figure 9: **Scene-level qualitative comparisons between DreamFusion, ProlificDreamer, and SteinDreamer w/ depth estimator.** Compared to existing methods, SteinDreamer presents more realistic textures with better details.

generation in Fig. 11. Similarly, it validates that score distillation's performance is highly correlated with variance of stochastic gradients.

Additionally, while we observe that SteinDreamer tends to exhibit better geometric consistency, mitigating broken geometries and ghosting artifacts, it comes at the cost that the produced texture might be smoother (and perhaps less rich) compared to ProlificDreamer. This phenomenon aligns with the underlying variance control mechanism: low-variance updates converge to more stable minima, whereas high-variance updates yield more diverse results. The strength of the control variates need to be carefully tuned to avoid over-smoothing.

D.3 More Quantitative Results

In addition to the overall quantitative comparison in Sec. 5, we also provide a breakdown table of the CLIP distance for every demonstrated object in this paper. The numerical evaluation of these results is reported in Tab. 5. Our observations indicate that SteinDreamer consistently outperforms all other methods, which improves CLIP score by ˜0.5 over ProlificDreamer. We provide a list that shows other prompts involved into quantitative evaluation for object generation in Tab. 6.

We also aim to empirically justify the argument that a depth estimator can induce a higher-correlated control variate for variance reduction than artisanal choices in SDS and VSD. To be more specific, we measure the correlation between the control variates $h_{SDS}, h_{VSD}, h_{SSD}$ and the target score function $f(t, \boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{c})$ (see Eq. 3). The results reported in Tab. 4 support our argument that control variates constructed with pre-trained models are more aligned with diffusion model predicted scores and can exhibit higher correlation than other choices.

Control Variates	h_{SDS}	h_{VSD}	h_{SSD} w/ Depth Pred.	h_{SSD} w/ CLIP
Correlation	0.121	0.370	0.514	0.477

Table 4: **Correlation of control variates.** The estimated correlation between the control variates and the distilled score function.

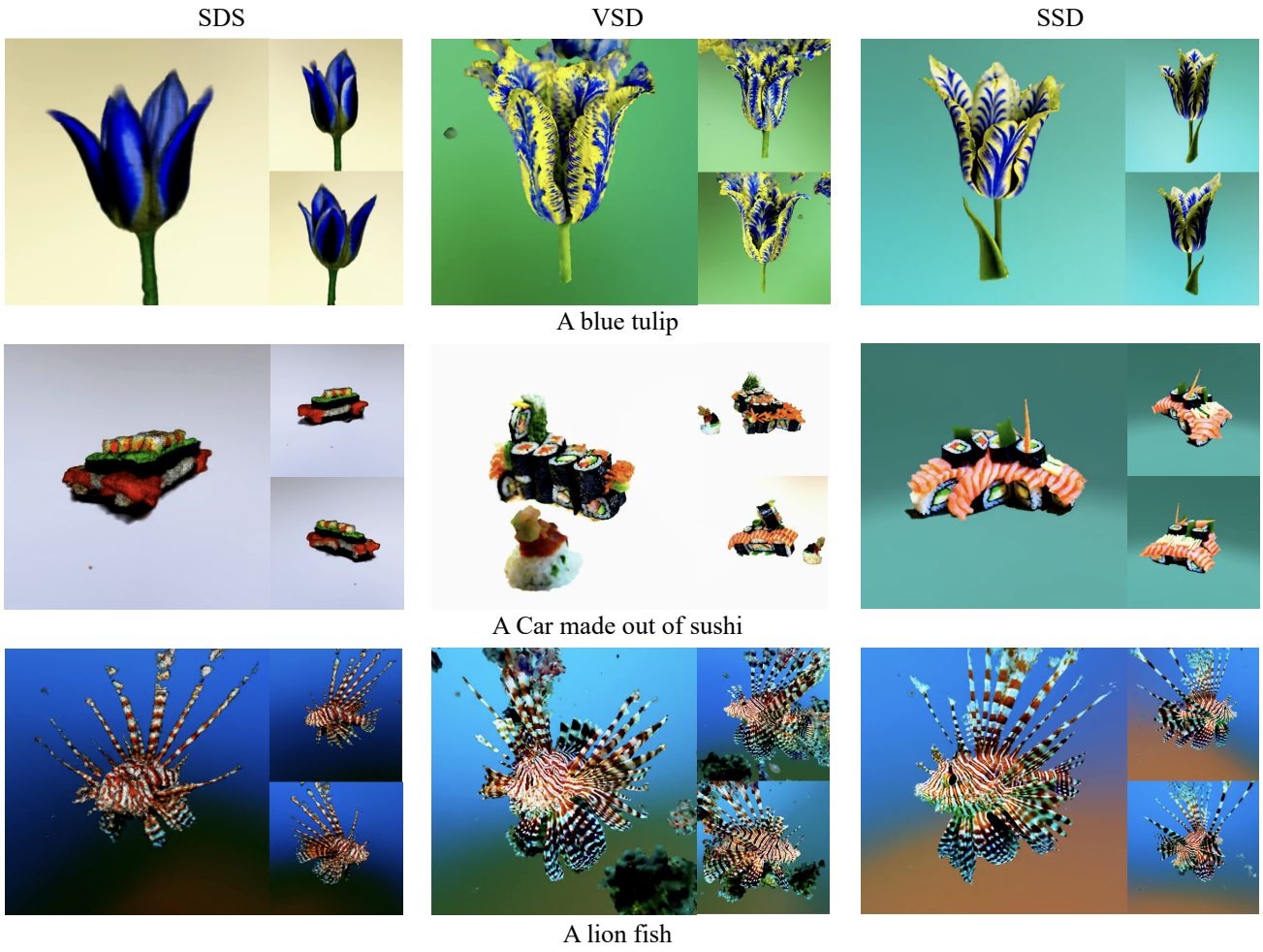

Figure 10: **Object-level qualitative comparisons.** Compared to existing methods, our SteinDreamer w/ depth estimator delivers smoother geometry, more detailed texture, and fewer floater artifacts.

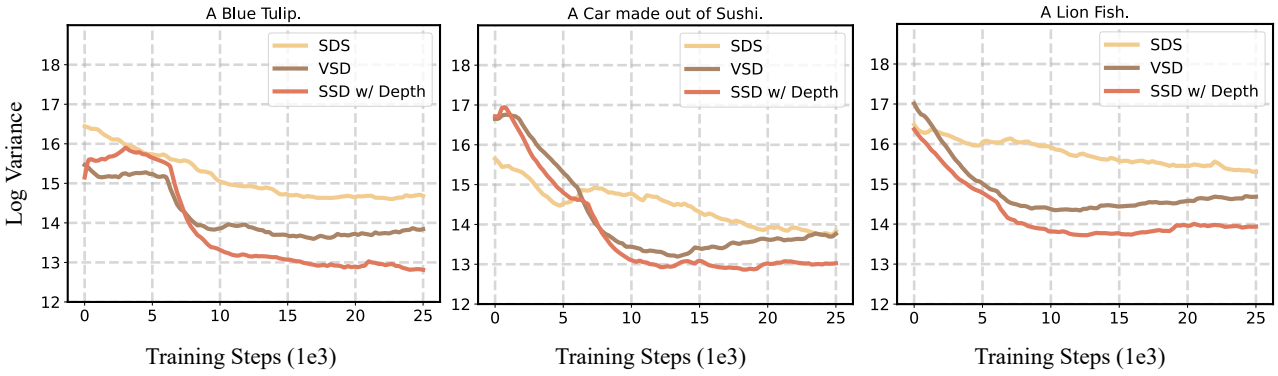

Figure 11: **Variance comparison of Δ_{SDS}, Δ_{VSD}, and Δ_{SSD} with depth estimator.** We visualize how the variance of the investigated three methods for every 1,000 steps. The variance decays as the training converges while Δ_{SSD} consistently achieves lower variance throughout the whole process.

D.4 Wall-Clock Time Benchmarking

In addition to faster convergence, we also test per-iteration wall-clock time for all methods. Results are listed in Tab. 7. The reported numbers are obtained by averaging the running time of corresponding methods with

Methods	"blue tulip"	"sushi car"	"lionfish"
SDS (Poole et al., 2022)	0.777	0.862	0.751
VSD (Wang et al., 2023c)	0.751	0.835	0.749
SSD w/ Depth (Ours)	0.734	0.754	0.735
	"cheesecake castle"	"dragon toy"	"dog statue"
SDS (Poole et al., 2022)	0.902	0.904	0.789
VSD (Wang et al., 2023c)	0.843	0.852	0.775
SSD w/ Normal (Ours)	0.794	0.806	0.751

Table 5: **Breakdown table.** We compare the CLIP distance (↓ the lower the better) of demonstrated results among different approaches. Best results are marked in color. Prompts: "blue tulip" is short for "a blue tulip", "sushi car" for "a car made out of sushi", "lionfish" for "a lionfish", "cheesecake castle" for "a Matte painting of a castle made of cheesecake surrounded by a moat made of ice cream", "dragon toy" for "a plush dragon toy", and "dog statue" for "michelangelo style statue of dog reading news on a cellphone".

a pineapple
a 3D model of an adorable cottage with a thatched roof
an elephant skull
a plate piled high with chocolate chip cookies
michelangelo style statue of dog reading news on a cellphone
a chimpanzee dressed like Henry VIII king of England
a delicious croissant
a sliced loaf of fresh bread
a small saguaro cactus planted in a clay pot
a blue tulip
a plate of fried chicken and waffles with maple syrup on them
a cauldron full of gold coins
a rabbit, animated movie character, high detail 3d model
a lionfish
a car made out of sushi
a DSLR photo of an imperial state crown of England
a rotary telephone carved out of wood
a marble bust of a mouse
a typewriter
a Matte painting of a castle made of cheesecake surrounded by a moat made of ice cream
a plush dragon toy
a praying mantis wearing roller

Table 6: **A list of text prompts.** All text prompts are collected from Wang et al. (2023c).

Methods	Sec. / Iter.
SDS (Poole et al., 2022)	1.063 ± 0.002
VSD (Wang et al., 2023c)	1.550 ± 0.004
SSD w/ Depth (Ours)	1.093 ± 0.005
SSD w/ Normal (Ours)	1.087 ± 0.004

Table 7: **Benchmarking wall-clock time**. We measure wall-clock time (seconds per iteration) for all considered methods. All reported timings are acquired on the same GPU device for a fair comparison.

six prompts in Tab. 5 for 10k iterations *on the same device*. In summary, SteinDreamer exhibits comparable per-iteration speed to SDS while significantly outperforming VSD in terms of speed. The trainable component $\boldsymbol{\mu}$ in SSD comprises only thousands of parameters, which minimally increases computational overhead and becomes much more efficient than tuning a LoRA in VSD. Notably, given that SSD can reach comparable visual quality in fewer steps, SteinDreamer achieves significant time savings for 3D score distillation.

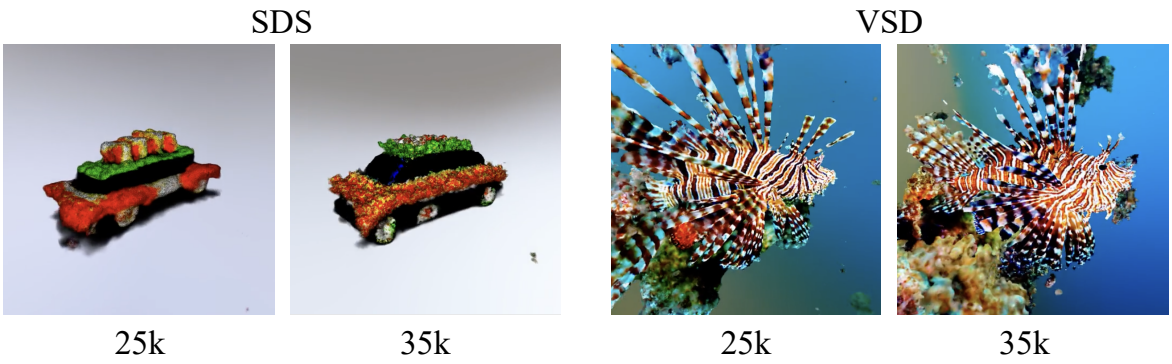

Figure 12: **Longer Training Results.** We train high-variance score distillation approaches SDS and VSD for extra 10k steps. Prompts: " car made out of sush" for SDS and "a lionfish" for VSD

D.5 Longer Training for Baselines

A naive solution to achieve better convergence with high-variance gradient descent is to increase training steps. We test this hypothesis in this section by training SDS and VSD on two scenes with 10k more steps. Qualitative results are presented in Fig. 12. We notice that longer training time cannot guarantee better convergence. We also quantitatively find that more optimization steps have negligible influence on the final CLIP scores, which float between 0.84 ˜0.86 for the prompt " car made out of sush" and 0.74 ˜0.75 for the prompt "a lionfish".

In optimization theory, variance plays a crucial role in determining the convergence rate of SGD algorithms (Garrigos and Gower, 2023) With a finite number of optimization steps and a standard learning rate, maintaining low variance is pivotal to ensure convergence. Training with noisy gradients introduces high instability, potentially resulting in a suboptimal solution or even divergence, as illustrated in Fig. 9.