Flow Score Distillation for Diverse Text-to-3D Generation

Runjie Yan[1], Kailu Wu[2], and Kaisheng Ma[2⋆]

[1] Institute for Interdisciplinary Information Sciences, Tsinghua University
[2] ArchipLab, Institute for Interdisciplinary Information Sciences, Tsinghua University
{yanrj21, wkl22}@mails.tsinghua.edu.cn
kaisheng@mail.tsinghua.edu.cn

Abstract. Recent advancements in Text-to-3D generation have yielded remarkable progress, particularly through methods that rely on Score Distillation Sampling (SDS). While SDS exhibits the capability to create impressive 3D assets, it is hindered by its inherent maximum-likelihood-seeking essence, resulting in limited diversity in generation outcomes. In this paper, we discover that the Denoise Diffusion Implicit Models (DDIM) generation process (*i.e.* PF-ODE) can be succinctly expressed using an analogue of SDS loss. One step further, one can see SDS as a generalized DDIM generation process. Following this insight, we show that the noise sampling strategy in the noise addition stage significantly restricts the diversity of generation results. To address this limitation, we present an innovative noise sampling approach and introduce a novel text-to-3D method called Flow Score Distillation (FSD). Our validation experiments across various text-to-image Diffusion Models demonstrate that FSD substantially enhances generation diversity without compromising quality.

Keywords: 3D Generation · Noise Prior · Diffusion Probability Flow ODE · Probability Density Distillation

1 Introduction

In the realm of 3D content creation, a crucial step within the modern game and media industry involves crafting intricate 3D assets. Recently, 3D generation has facilitated the creation of 3D assets with ease. 3D generative models could be trained directly on certain representations (*e.g.* point clouds [1, 27], voxel [41, 53] and mesh [57]). However, despite the recent efforts of Ojaverse [6], 3D data remains relatively scarce, especially when compared to the abundant 2D image data available on the internet. This scarcity constrains the generative capabilities of models trained solely on 3D datasets. Notably, the most prevailing text-to-3D approach is based on Score Distillation Sampling (SDS), proposed by Dreamfusion [32] and SJC [47]. SDS effectively tackles the scarcity of 3D data

⋆ Corresponding author

by leveraging pretrained 2D text-to-image Diffusion Models, without directly training models on 3D datasets.

Fig. 1: Generation results of FSD and baseline method SDS. FSD uses pretrained text-to-image Diffusion Models to generate realistic 3D models from text prompts. We improve the noise sampling strategy upon SDS and achieve **diverse generation results without compromising quality**.

SDS is designed to optimize any representations (*e.g.* Neural Radiance Field [29, 30, 49], 3D Gaussian Splatting [18], Mesh [20, 39] or even 2D images) that could render 2D images through probability density distillation [31] using the learned score functions from the Diffusion Models. One of the main limitations of current SDS-based methods is that their distillation objectives will maximize

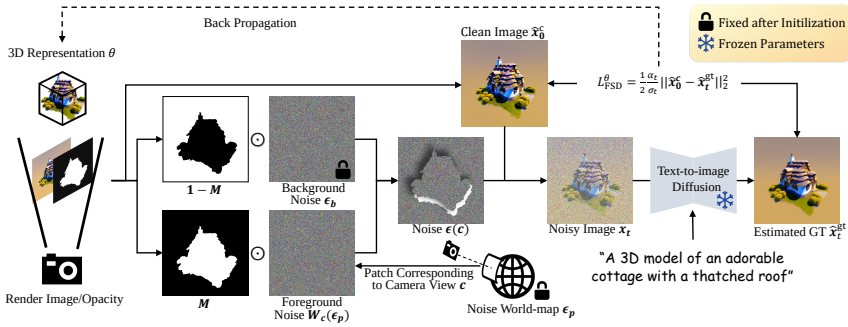


Fig. 2: Methods overview of FSD. We propose Flow Score Distillation for text-to-3D generation by lifting a pretrained Diffusion Model. FSD renders an image $\boldsymbol{g}_\theta(\boldsymbol{c})$ from the 3D representation and adds noise $\boldsymbol{\epsilon}(\boldsymbol{c})$ to the rendered image. To compute parameter updates according to L_{FSD}^{θ}, FSD uses a frozen text-to-image Diffusion Model to predict the noise $\boldsymbol{\epsilon}(\boldsymbol{c})$ added on image $\boldsymbol{g}_\theta(\boldsymbol{c})$. Similar to SDS [32,47], FSD computes L_{FSD}^{θ} by an image reconstruction loss between the "clean image" $\hat{\boldsymbol{x}}_t^c = \boldsymbol{g}_\theta(\boldsymbol{c})$ and "ground-truth image" $\hat{\boldsymbol{x}}_0$ predicted by the pretrained Diffusion Model. FSD further adopts timestep annealing schedule and noise sampling strategy. Instead of sampling noise from Gaussian distribution at each step of the optimization like SDS, we generate noise according to the deterministic noise function $\boldsymbol{\epsilon}(\boldsymbol{c})$, which is determined at the beginning of the optimization.

the likelihood of the image rendered from the 3D representations, which leads to limited diversity. Additionally, The success of SDS relies heavily on large classifier-free guidance (CFG) [13,56]. However, this reliance introduces issues like over-saturation and further restricts the diversity of the generation results. Despite several subsequent efforts [2,15,17,21,46,48,50,58] to enhance SDS, the maximum-likelihood-seeking essence of the method remains unchanged. Notably, ProlificDreamer [50] introduces Variational Score Distillation (VSD) and uses a fine-tuned Diffusion Model to model distribution on particles, which could alleviate the maximum-likelihood-seeking issues. However, training costs could grow linearly with the particle number of VSD. ESD [48] also points out that single-particle VSD is equivalent to SDS, which remains rooted in the essence of maximum likelihood seeking.

In this paper, we present a fresh perspective on SDS by viewing it as a generalized DDIM [43] generation process for 3D representations. Specifically, we discovered that the DDIM generation process (*i.e.* PF-ODE [45]) can be succinctly expressed using an analogue of SDS loss. Surprisingly, by studying the difference between the analogue of SDS loss and the original form of SDS loss, we find that the noise sampling strategy during the noise addition stage appears to be the main cause that drives SDS toward mode-seeking behavior. SDS-based methods typically use random noise sampled from a Gaussian distribution at each optimization step, following the proposal of Dreamfusion [32] and SJC [45]. However, the variation in sampled noise can lead to varied optimization

directions, which may harm the performance of SDS, as observed by ISM [21]. As we will show in this work, PF-ODE can be expressed by an analogue of SDS loss that uses the same noise throughout the generation process, which is different from the original proposal of SDS [32, 47]. Based on this insight, we propose a novel noise sampling strategy to align SDS with the DDIM generation process on 3D representations.

This paper aims to overcome the aforementioned diversity challenge of SDS. We will first reveal an underlying connection between SDS and DDIM on 2D image generation. We will also show that the noise sampling strategy could be the primary factor that leads to the restricted diversity. Based on this insight, we will give our interpretation of SDS. From our novel viewpoint on SDS, we propose a novel approach called *Flow Score Distillation* (FSD). FSD improves SDS by using a carefully designed noise sampling strategy. We lift our observations on image generation with FSD to 3D by proposing a view-dependent noise function $\boldsymbol{\epsilon}(\boldsymbol{c})$. We conduct validation experiments across various 2D Diffusion Models and demonstrate that FSD can achieve diverse generation outcomes without compromising quality as well as introducing extra training costs. Overall, our contributions can be summarized as follows.

- We provide an in-depth analysis of SDS, an effective method in text-to-3D generation. We present a proposition that reveals the underlying connection between SDS and DDIM. Specifically, the DDIM generation process (*i.e.* PF-ODE) can be succinctly expressed using an analogue of SDS loss. As a result, we can interpret SDS as a generalized DDIM generation process on 3D representations.
- Building upon our new insight into SDS, we identify that the diversity degradation stems from the noise sampling strategy. We show that adding the same noise throughout the generative process can enhance both generation quality and diversity in 2D generation with SDS. Hence, we show the noise sampling strategy is the primary factor that restricts the diversity.
- We introduce FSD as a cheap but effective solution to tackle the diversity challenges arising from the maximum-likelihood-seeking nature of SDS. We propose a deterministic *world-map noise function* to generate coarsely aligned noise in 3D space. By applying a reasonable noise sampling strategy, FSD breaks free from the mode-seeking nature of SDS-like methods. Our validation experiments demonstrate that FSD substantially enhances generation diversity without compromising quality.

2 Preliminaries and Related Works

In this section, we will provide preliminaries on Diffusion Models, differential equations associated with them, and SDS. Additionally, we will also introduce works related to this paper. Our notation conventions mainly follow Song *et al.* [45], Lu *et al.* [26] and Poole *et al.* [32].

2.1 Diffusion Models

Diffusion Models [12,42,45] are a family of powerful generative models that are trained to gradually transform Gaussian noise to samples from a target distribution p_0. Their generation ability is further enhanced given datasets comprising billions of image-text pairs [4,37,38].

Assume the target distribution is p_0 and condition y. Diffusion Models define a forward process $\{x_t\}_{t\in[0,T]}$ starting from $\boldsymbol{x}_0 \sim p_0(\cdot|y)$, such that for any $t \in [0,T]$ the distribution of x_t conditioned on x_0 satisfies:

$$\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}), \quad \boldsymbol{x}_0 \sim p_0(\cdot|y), \tag{1}$$

where $\alpha_t, \sigma_t \in \mathbb{R}^+$ are functions of t, defined by the *noise schedule* of the Diffusion Model. And the (noisy) distribution at timestep t is noted as p_t.

In practice, a Diffusion Model is a neural network $\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t|y,t)$ parameterized by ϕ and is trained by minimizing the following objective [43,45]:

$$L^{\phi}_{\text{DMs}} = \frac{1}{2}\mathbb{E}_{\boldsymbol{x}_0\sim p_0(\cdot|y),\boldsymbol{\epsilon},t}\left[w_t||\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t|y,t) - \boldsymbol{\epsilon}||_2^2\right], \tag{2}$$

where w_t is a weighting function. As the training objective implies, a Diffusion Model can be seen as predicting the Gaussian noise added to the clean data $\boldsymbol{x}_0 \sim p_0(\cdot|y)$. Song *et al.* [45] proved that:

$$\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t|y,t) \approx -\sigma_t \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}_t|y), \tag{3}$$

if the Diffusion Model $\boldsymbol{\epsilon}_\phi$ is trained to almost optimum. The term $\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}_t|y)$ in the above equation is also known as the *score function*.

2.2 Diffusion PF-ODE and DDIM

To generate samples from Diffusion Models, there exist different methods among the Diffusion Models family. Denoise Diffusion Implicit Models (DDIM) [43] designed a deterministic method for fast sampling from Diffusion Models. Recent works [16,26,36] have shown that the sampling algorithm of DDIM is a first-order discretization of the Diffusion Probability Flow Ordinary Differential Equation (PF-ODE) [45]. Therefore, in this work, we will not make specific distinction between DDIM generation process and PF-ODE.

Theoretically, Diffusion PF-ODE yields the same marginal distribution as the forward process of Diffusion Models (Eq. (1)) [45]. We can write Diffusion PF-ODE [16,45] (see detailed derivation in Appendix) as:

$$\frac{\mathrm{d}(\boldsymbol{x}_t/\alpha_t)}{\mathrm{d}t} = \frac{\mathrm{d}(\sigma_t/\alpha_t)}{\mathrm{d}t}\left(-\sigma_t \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}_t|y)\right) \tag{4}$$

$$= \frac{\mathrm{d}(\sigma_t/\alpha_t)}{\mathrm{d}t}\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t|y,t), \quad \boldsymbol{x}_T \sim p_T(\boldsymbol{x}_T|y). \tag{5}$$

Notably, one can generate a sample $\boldsymbol{x}_0$ in the target distribution $p_0(\boldsymbol{x}_0|y)$ by following the PF-ODE trajectory from $t = T$ to $t = 0$, starting from $\boldsymbol{x}_T \sim p_T(\boldsymbol{x}_T|y) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

2.3 Score Distillation Sampling

Recently, DreamFusion [32] and SJC [47] proposed Score Distillation Sampling (SDS) to generate 3D models by optimizing a differentiable 3D representation using priors from text-to-image Diffusion Models. Follow-up works tried to improve upon SDS through various aspects, *e.g.* coarse-to-fine training strategy [5,22,50], disentangled 2D-3D priors [5,28,50] and refined formulas [2,21,46,48,50,56,58]. Moreover, due to the lack of comprehensive 3D-aware knowledge, multi-face Janus problem often arises when using SDS [32]. To mitigate this challenge, one can consider replacing the text-to-image Diffusion Models with Diffusion Models designed for object novel view synthesis [23–25, 51, 55] or multi-view Diffusion Models [40]. Such an adaptation can significantly alleviate the multi-face Janus problem encountered in 3D generation using SDS.

SDS is first introduced by DreamFusion [32] and SJC [47] to apply image diffusion priors for 3D generation. SDS can optimize on any representations parameterized by θ, which can render an image $\boldsymbol{g}_\theta(\boldsymbol{c})$, given camera parameter $\boldsymbol{c}$. Basically, SDS defines a probability density distillation [31] loss, denoted as L^θ_{SDS}, whose gradient writes as follows:

$$\nabla_\theta L^\theta_{\text{SDS}} = \mathbb{E}_{\boldsymbol{c},t}\left[w_t \frac{\sigma_t}{\alpha_t} \nabla_\theta D_{\text{KL}}\left(p_t(\boldsymbol{x}_t|\boldsymbol{x}_0 = \boldsymbol{g}_\theta(\boldsymbol{c}))||p_t(\boldsymbol{x}_t|y)\right)\right] \tag{6}$$

$$= \mathbb{E}_{\boldsymbol{\epsilon},\boldsymbol{c},t}\left[w_t\left(\boldsymbol{\epsilon}_\phi(x_t|y,t) - \boldsymbol{\epsilon}\right)\frac{\partial \boldsymbol{g}_\theta(\boldsymbol{c})}{\partial \theta}\right], \tag{7}$$

where ϵ_ϕ is a text-to-image Diffusion Model and y is the generation condition, *e.g.* text prompts. In order to generate 3D content, SDS needs to go through an optimization process on the 3D representation parameter θ to generate a single 3D model.

Even though SDS can produce high-fidelity objects, there has been ongoing debate about the underlying theory. Recent works [21, 40] also show that L^θ_{SDS} is equivalent to a reconstruction loss:

$$L^\theta_{\text{SDS}} = \mathbb{E}_{\boldsymbol{\epsilon},\boldsymbol{c},t}\left[\frac{1}{2} w_t \frac{\alpha_t}{\sigma_t} ||\hat{\boldsymbol{x}}^{\text{c}}_t - \hat{\boldsymbol{x}}^{\text{gt}}_t||^2_2\right], \tag{8}$$

where $\hat{\boldsymbol{x}}^{\text{c}}_t = \boldsymbol{g}_\theta(\boldsymbol{c})$ and $\hat{\boldsymbol{x}}^{\text{gt}}_t = \frac{\boldsymbol{x}_t - \sigma_t \boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t|y,t)}{\alpha_t}$ is the one-step "estimated ground-truth image" from Diffusion Models, whose gradient is detached. Some other works [17, 46, 56] also tried to explain SDS by analyzing the function of each component of SDS loss. In this paper, we will provide another interpretation of SDS: it can be viewed as a generalized DDIM generation process.

Another simple yet effective technique is to apply timestep annealing trick [15,50,58], which can improve generation quality significantly. This technique is intuitive because reducing added noise during the latter stages of optimization, enabling models to discern finer details and iteratively improve upon them. Let us denote the time in the SDS optimization process as τ and the term that was taken expectation in the definition of SDS (Eq. (8)) as $L^\theta_{\text{sds}}(\boldsymbol{\epsilon}, \boldsymbol{c}, t) = \frac{1}{2}\frac{\alpha_t}{\sigma_t}||\hat{\boldsymbol{x}}^{\text{c}}_t - \hat{\boldsymbol{x}}^{\text{gt}}_t||^2_2$.

Fig. 3: Generation results of different methods on image space with the same random seeds. FSD can generate images that are very similar to images generated by DDIM given the same initial noise (implied by Proposition 1). However, FSD can also be used for 3D generation, a task for which DDIM is not suitable. See experiment details in Appendix.

We use lowercase letters footnote for L^{θ}_{sds} to distinguish it from Eq. (8). Finally, the SDS optimization process with timestep annealing can be written as:

$$\frac{\mathrm{d}\theta}{\mathrm{d}\tau} = w_t \mathbb{E}_{\boldsymbol{\epsilon},\boldsymbol{c}} \left[\nabla_\theta L^{\theta}_{\text{sds}}(\boldsymbol{\epsilon}, \boldsymbol{c}, t = t(\tau)) \right], \tag{9}$$

where $t(\tau)$ is a monotonically decreasing function of τ.

3 Flow Score Distillation for 2D Generation

In this section, we only consider generation on 2D using SDS as SDS loss can also be applied to image representations. In this case, $\theta = \boldsymbol{g}_\theta(\boldsymbol{c})$ and $\frac{\partial \boldsymbol{g}_\theta(\boldsymbol{c})}{\partial \theta} = \boldsymbol{I}$. Then L^{θ}_{sds} becomes the following form:

$$\nabla_\theta L^{\theta}_{\text{sds-2d}}(\boldsymbol{\epsilon}, t) = \boldsymbol{\epsilon}_\phi(x_t | y, t) - \boldsymbol{\epsilon}. \tag{10}$$

We will reveal a simple but profound connection between SDS and DDIM and give our interpretation of SDS in this section.

3.1 Simplified Formulation of Diffusion PF-ODE

We first reveal that PF-ODE (Eq. (5)) can be formulated by an analogue of SDS (Eq. (10)) in this section. We first define

$$\boldsymbol{x}_t = \alpha_t \hat{\boldsymbol{x}}^{\text{c}}_t + \sigma_t \tilde{\boldsymbol{\epsilon}}, \tag{11}$$

where $\tilde{\boldsymbol{\epsilon}}$ is a constant for each ODE trajectory. Notice that when $t = T$, the initial condition of the ODE gives $\tilde{\boldsymbol{\epsilon}} = 0 \cdot \hat{\boldsymbol{x}}_t^{\mathrm{c}} + 1 \cdot \tilde{\boldsymbol{\epsilon}} = \boldsymbol{x}_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Intuitively, **$\tilde{\boldsymbol{\epsilon}}$ can be viewed as the noise added to $\hat{\boldsymbol{x}}_t^{\mathbf{c}}$, and $\hat{\boldsymbol{x}}_t^{\mathbf{c}}$ as the clean image at timestep** t. So we will also refer to $\tilde{\boldsymbol{\epsilon}}$ as *the initial noise* apart from *the added noise* in this paper. It is noteworthy that the concept of the clean image $\hat{\boldsymbol{x}}_t^{\mathrm{c}} = \frac{\boldsymbol{x}_t - \sigma_t \tilde{\boldsymbol{\epsilon}}}{\alpha_t}$ is different from the aforementioned estimated ground-truth image $\hat{\boldsymbol{x}}_t^{\mathrm{gt}} = \frac{\boldsymbol{x}_t - \sigma_t \boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t | y, t)}{\alpha_t}$. By applying "*change-of-variable*" trick and change the variable of the Diffusion PF-ODE from $\boldsymbol{x}_t$ to $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$, we have:

Proposition 1 (An equivalent form of Diffusion PF-ODE). *Diffusion PF-ODE (Eq. (5)) can be equivalently formulated by an analogue of SDS loss (Eq. (10)):*

$$\frac{\mathrm{d}\hat{\boldsymbol{x}}_t^c}{\mathrm{d}t} = \frac{\mathrm{d}(\sigma_t/\alpha_t)}{\mathrm{d}t} \left[\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t | t, y) - \tilde{\boldsymbol{\epsilon}}\right] \tag{12}$$

$$= w_t' \nabla_\theta L_{sds\text{-}2d}^\theta(\tilde{\boldsymbol{\epsilon}}, t), \tag{13}$$

where $\boldsymbol{x}_t = \alpha_t \hat{\boldsymbol{x}}_t^c + \sigma_t \tilde{\boldsymbol{\epsilon}}$, $\theta = \hat{\boldsymbol{x}}_t^c$ *and* $w_t' = \frac{\mathrm{d}(\sigma_t/\alpha_t)}{\mathrm{d}t}$ *is a weighting scalar.*

Please refer to Appendix for detailed derivation of this proposition. Remarkably, in the context of image generation, we observe that the evolution direction of PF-ODE aligns precisely with the gradient of the SDS loss (Eq. (10)).

3.2 Flow Score Distillation on 2D

Even though we found the evolution direction of Diffusion PF-ODE (Eq. (12)) is very samilar to the SDS loss, there exist some notable differences compared to the original definition of SDS loss (Eq. (6)). Specifically, i) the timestep in a DDIM process is monotonically decreasing, aligning with the timestep annealing technique [15, 50, 58]. But SDS uses randomly sampled timestep. ii) And the change-of-variable trick (Eq. (11)) we used during our simplification process implies we should also add the same noise $\tilde{\boldsymbol{\epsilon}}$ throughout the SDS generation process, to align it with DDIM. In contrast, the original SDS uses uncorrelated random noise.

As we will demonstrate in subsequent sections and through our experiments, the second difference between DDIM and SDS significantly influences generation diversity. Therefore, we term our approach that combines **timestep annealing and consistent noise sampling strategy throughout the generation process** as *Flow Score Distillation* (FSD) to differ it from SDS. We visualize image generation results using several SDS-like methods, FSD and DDIM in Fig. 3 to demonstrate the differences between SDS-based methods and FSD.

3.3 Analysis of the Noise Sampling Strategy

Our empirical investigation reveals that the generation results produced by SDS exhibit an undesirable tendency toward over-smoothness and lack of diversity.

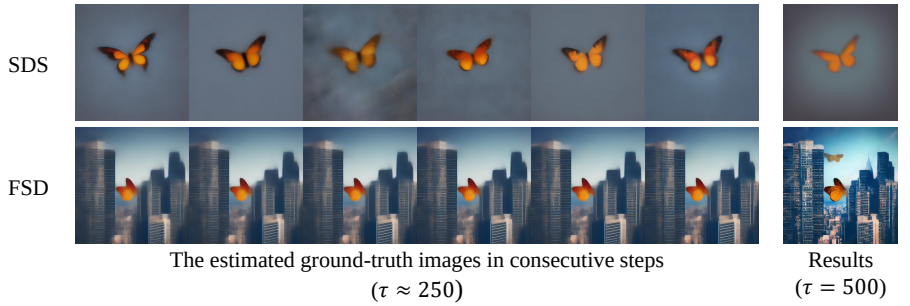


Fig. 4: Visualization of FSD and SDS for image generation. We visualize generation results and the estimated ground-truth images in consecutive steps at halfway of the generation for both FSD and SDS [32, 45]. We find the estimated ground-truth images of FSD are consistent, while the contents vary greatly in the ground-truth images of SDS. We set CFG=7.5 and adopted the same linear timestep annealing schedule for both FSD and SDS in the experiment for this figure.

Remarkably, even with the timestep annealing technique, this issue persists. This tendency originates from the maximum-likelihood-seeking nature implied by its definition (Eq. (6)) where it models the current distribution using a Dirichlet function centered at $\boldsymbol{g}_\theta(\boldsymbol{c})$. In contrast, not only does FSD yield diverse outcomes, but it can also generate highly detailed samples. This can be attributed to that FSD is more aligned with a DDIM process, which can generate samples from exactly the target distribution p_0. So we conclude that the noise sampling strategy can affect the generation diversity greatly.

To gain an intuitive understanding of these effects, we also explain the reasons by following the reconstruction loss interpretation [21, 40]. SDS loss can be seen as the reconstruction error between the rendered images and the one-step estimated ground-truth images predicted by the Diffusion Models. When different noise is added, the generated ground-truth images differ greatly (Demonstrated in Fig. 4), as noted by ISM [21]. As a result, SDS can only generate averaged outcomes. In contrast, applying identical noise at each optimization step ensures a more consistent optimization trajectory. So FSD can generate diverse results.

3.4 Compare FSD with SDS and DDIM on 2D

We summarize the difference between FSD, SDS, and DDIM when applied to 2D images as follows:

Noise Sampling Strategy Both FSD and DDIM add fixed noise sampled from $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ at the beginning of optimization to the clean image $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$. While SDS adds random noise sampled from $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ at each step of the optimization to the clean image $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$.

Fig. 5: Impact of initial noise $\tilde{\epsilon}$. Experiments show that the local textures of noise added during FSD optimization are highly correlated with the textures of the final image. We shuffle the patches of initial noise $\tilde{\boldsymbol{\epsilon}}$ used by FSD and observe that the textures of generated images are shuffled in the same way. This property inspired our design of world-map noise function $\boldsymbol{\epsilon}(\boldsymbol{c})$ for 3D generation in this work. In this figure, the parts framed by dotted lines of the same color share the same initial noise $\tilde{\boldsymbol{\epsilon}}$ patches.

Optimizer Both FSD and SDS use Adam [19] (or Adan [54]) to update the clean image parameter $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$. Moreover, the impact of the weighting parameter w_t is negligible as long as it changes slowly over t since Adam optimizer is invariant to the scale of the gradients [19]. While DDIM updates on $\boldsymbol{x}_t$ directly according to the first-order discretization of PF-ODE.

Diffusion Timestep Schedule FSD tries to model the schedule of DDIM by using a monotonically decreasing function $t(\tau)$. While SDS uses time schedule which samples t uniformly. Moreover, the convergence time τ_{end} of FSD and SDS is hard to determine theoretically, but DDIM will converge when $t = T$.

4 Lifting Flow Score Distillation to 3D

As highlighted in Sec. 3.3, we have identified that the noise sampling strategy might contribute to the decline of the diversity of SDS significantly. Building upon this insight, we follow the discussion of FSD in Sec. 3.2 and propose to use deterministic noise generation strategy. We can directly generalize FSD to arbitrary 3D representations $\boldsymbol{g}_\theta(\boldsymbol{c})$:

$$\nabla_\theta L^\theta_{\mathrm{FSD}} = \mathbb{E}_{\boldsymbol{c}} \left[\nabla_\theta L^\theta_{\mathrm{sds}}(\boldsymbol{\epsilon} = \boldsymbol{\epsilon}(\boldsymbol{c}), \boldsymbol{c}, t = t(\tau)) \right] \tag{14}$$

$$= \mathbb{E}_{\boldsymbol{c}} \left[\left(\boldsymbol{\epsilon}_\phi(x_t | y, t(\tau)) - \boldsymbol{\epsilon}(\boldsymbol{c}) \right) \frac{\partial \boldsymbol{g}_\theta(\boldsymbol{c})}{\partial \theta} \right], \tag{15}$$

where $\boldsymbol{x}_t = \alpha_t \boldsymbol{g}_\theta(\boldsymbol{c}) + \sigma_t \boldsymbol{\epsilon}(\boldsymbol{c})$, $\boldsymbol{\epsilon}(\boldsymbol{c})$ is a deterministic noise function generated at the beginning of the optimization and $t(\tau)$ is a monotonically decreasing timestep schedule function to optimization time τ. We do not specify the form of the deterministic noise function $\boldsymbol{\epsilon}(\boldsymbol{c})$ in FSD. However, we propose some rules for designing $\boldsymbol{\epsilon}(\boldsymbol{c})$ based on the actual generation effect in Appendix, according to our practical experiences. We will also introduce the *world-map noise function* as $\boldsymbol{\epsilon}(\boldsymbol{c})$ for our experiments of 3D generation with FSD in this paper.

Alternatively, FSD loss can be seen as applying DDIM generation process on 3D representations through Jacobian of a differentiable renderer. This viewpoint

shares some similarities to the interpretation of SDS from SJC [47], who consider SDS as back-propagating the score [45] of Diffusion Models through Jocabian of the renderer. Meanwhile, $\boldsymbol{\epsilon}(\boldsymbol{c})$ is a deterministic noise function that generates correlated noise between views, which aligns with the prior that the nearby views are correlated. The design of FSD ensures the optimization directions are consistent, particularly when similar $\boldsymbol{c}$s are sampled. As the $\boldsymbol{\epsilon}(\boldsymbol{c})$s are similar, the generated ground-truth images should be consistent as well. Notably, recent works [3,7,33] on video generation also find using designed video noise prior can improve the capabilities of Video Diffusion Models.

4.1 Designing $\boldsymbol{\epsilon}(\boldsymbol{c})$.

Failure of a Vanilla Design of $\boldsymbol{\epsilon}(\boldsymbol{c})$ A vanilla design of $\boldsymbol{\epsilon}(\boldsymbol{c})$ can be $\boldsymbol{\epsilon}(\boldsymbol{c}) = \boldsymbol{\epsilon}$, which is a constant function. However, according to our experiments on text-to-3D generation, such a design can lead to poor geometry of the generated samples. Typically, holes on the surfaces are observed (Fig. 6). We attribute this effect to the uneven convergence speed of FSD in 3D space caused by the constant noise function. Our experiments on 2D show that the local textures of noise added during FSD optimization are highly correlated with the local textures of the final image (Demonstrated in Fig. 5). And in text-to-3D generation with FSD, the generated ground-truth images have more consistent textures at the center point than other points in 3D space, due to the sampling strategy of camera view $\boldsymbol{c}$. As a result, the convergence speed at the center point is much higher than at other points, leading to holes on the surfaces. Flaws are also observed in Video Diffusion Models that adopt fixed noise prior, due to similar reasons, which is known as the textures sticking problem [3].

World-map Noise Function $\boldsymbol{\epsilon}(\boldsymbol{c})$. To avoid relating specific noise textures to specific points in 3D space throughout the generation process but still augment the consistency of added noise between camera views, we propose *world-map noise function* $\boldsymbol{\epsilon}(\boldsymbol{c})$ in this paper (visualized in Fig. 2), which aligns noise textures coarsely in 3D space while avoiding converging too fast at a specific point in 3D space (See discussion in Appendix).

Specifically, we first compute a foreground mask $\boldsymbol{M}(\boldsymbol{r}) = [\alpha(\boldsymbol{r}) > \alpha_0]$ for each ray $\boldsymbol{r}$, where $0 \leq \alpha(r), \alpha_0 \leq 1$ is the opacity of pixel $\boldsymbol{r}$. We add the same noise for the background and query a noise patch from the noise world-map $\boldsymbol{\epsilon}_p$ of size $D \times H \times W$ for the foreground. Let us denote the camera parameters sampled on the sphere as $\boldsymbol{c} = (\text{FOV}, r_{\text{cam}}, \theta_{\text{cam}}, \phi_{\text{cam}})$. Both $\boldsymbol{\epsilon}_b$ and $\boldsymbol{\epsilon}_p$ are sampled from $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ at the beginning of the optimization. And $\boldsymbol{\epsilon}$ is re-sampled from $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ at each step of the optimization. The world-map noise function $\boldsymbol{\epsilon}(\boldsymbol{c})$ is:

$$\boldsymbol{\epsilon}(\boldsymbol{c}) = \sqrt{\beta} \cdot \left((1 - \boldsymbol{M}) \odot \boldsymbol{\epsilon}_b + \boldsymbol{M} \odot \boldsymbol{W}_{(W\frac{\phi_{\text{cam}}}{2\pi}, H\frac{\theta_{\text{cam}}}{\pi})}(\boldsymbol{\epsilon}_p) \right) + \sqrt{1-\beta} \cdot \boldsymbol{\epsilon}, \quad (16)$$

where β is a blending factor to balance between random noise and deterministic noise. If not specified, we set $\beta = 1$ in this work so that $\boldsymbol{\epsilon}(\boldsymbol{c})$ is a completely

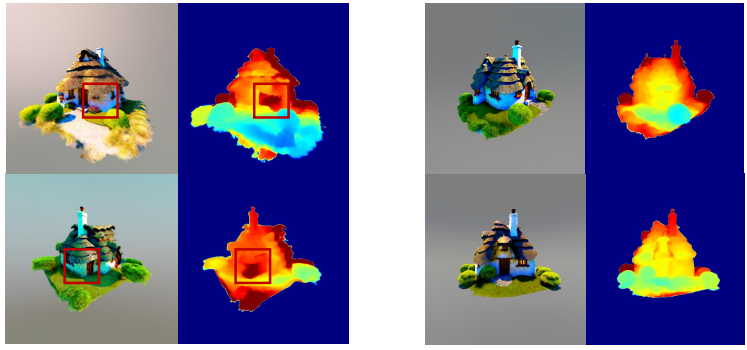


Fig. 6: Compare world-map noise function $\epsilon(c)$ with a vanilla design of constant function ϵ. We visualize the rendered images and depth maps for the two noise methods. The vanilla design of noise function ϵ can easily lead to holes on the surfaces (framed in red), even when the RGB images seem plausible. In contrast, no obvious flaws are observed in the results of FSD with the world-map noise function.

deterministic noise function. $\boldsymbol{W}_{(W\frac{\phi_{\text{cam}}}{2\pi}, H\frac{\theta_{\text{cam}}}{\pi})}(\boldsymbol{\epsilon}_p)$ operation refers to the noise patch of size $D \times H_{\text{hidden}} \times W_{\text{hidden}}$ centered at position $(W\frac{\theta_{\text{cam}}}{2\pi}, H\frac{\phi_{\text{cam}}}{\pi})$ on the noise worldmap $\boldsymbol{\epsilon}_p$. We also provide a pseudocode of our algorithm in Appendix.

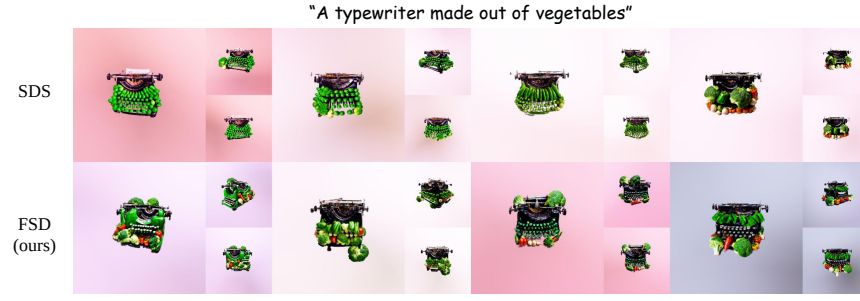
4.2 Compare FSD with SDS on 3D

Apart from timestep annealing trick [15, 50, 58], FSD is different from SDS in terms of noise sampling strategy as well. In case when the same camera view $\boldsymbol{c}$s are sampled, FSD yields consistent one-step estimated ground-truth images since $\boldsymbol{\epsilon}(\boldsymbol{c})$ is the same. Even when different camera views $\boldsymbol{c}$s are sampled, the ground-truth images are still coarsely aligned. In contrast, one can see SDS as using an uncorrelated noise prior $\boldsymbol{\epsilon}(\boldsymbol{c})$ on $\boldsymbol{c}$, which always yields ground-truth images that are inconsistent and have notable differences. We also discuss the differences between our method with recent works [8, 52] in Appendix B.

5 Experiments

5.1 Implementation Details

Unless otherwise specified, our experiments are conducted with the threestudio codebase [9]. We mainly conduct text-to-3D experiments using two different categories of pretrained Diffusion Models. Specifically, Stable Diffusion [43] that could generate a single image from the text prompt and MVDream [40] that could generate multi-view images of an object from the text prompt. We apply several tricks that we found helpful to improve generation quality **on both baselines and our method**. We use random seeds from 0 to 3 for each prompt



(a) Stable Diffusion [43] as backbone Diffusion Model

(b) MVDream [40] as backbone Diffusion Model

Fig. 7: Comparisons to baseline on text-to-3D Generaion. Our method can generate diverse 3D models with realistic and detailed appearances. We compare our method with the baseline using different pretrained Diffusion Models. Prompt for this figure: "A typewriter made out of vegetables". See additional experiments in Appendix.

by default to demonstrate the diversity of generated samples. Please refer to Appendix for more implementation details.

5.2 Evaluation on Text-to-3D Generation

Stable Diffusion. We visualize our experiment results of SDS [40,47] (baseline) and FSD in Fig. 7a using Stable Diffusion [43] (Fig. 7a). Due to single-image Diffusion Models' lack of 3D knowledge, both SDS and FSD may have multi-face Janus problem [32] and can not generate plausible results for all random seeds. So the generation results are usually cherry-picked in previous works, which may be potentially unfair. Instead of cherry-picking on generated samples, we manually pick some prompts on which SDS-like methods may not suffer from multi-face Janus problem in this experiment.

MVDream. We visualize our experiment results of SDS [40,47] (baseline) and FSD (ours) using MVDream [40] in Fig. 7b. From our practical observations, MVDream usually yields degraded generation results compared with Stable Diffusion [43] but seldomly suffers from multi-face Jauns problem. As a result of

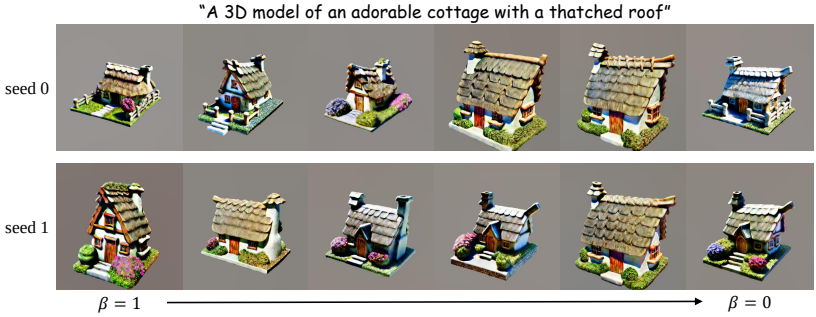


Fig. 8: Ablation on blending factor β. parameter β in the world-map noise function (Eq. (16)) is designed for balance between FSD ($\beta = 1$) and SDS ($\beta = 0$). We set $\beta = 1$ in the other part of this work and show results of FSD with different βs in this figure.

the degradation, text-to-3D generation results of MVDream are almost identical with different random seeds when using SDS. However, FSD can still generate diverse results with MVDream.

Quantitative Results. Please refer to Appendix.

5.3 Ablation Study

Ablation on blending factor β. β in Eq. (16) is the blending factor designed for balancing between the deterministic noise of FSD and the random noise of SDS. In the other part of this work, we fix $\beta = 1$, which means the noise function $\boldsymbol{\epsilon}(\boldsymbol{c})$ is completely deterministic. We show the impact of beta in Fig. 8.

Ablation on other hyperparameters in $\boldsymbol{\epsilon}(\boldsymbol{c})$. We use a parameter Θ to control the size (H and W) of noise world-map in Eq. (16). H and W are determined by Θ according to $H = H_{\text{hidden}} \frac{\pi}{\Theta}$ and $W = W_{\text{hidden}} \frac{2\pi}{\Theta}$. In practice, we found FSD are prone to the parameter Θ. See details in Appendix.

6 Conclusion

Conclusions. In this work, we systematically study the problem of text-to-3D generation. We first review the theorems of SDS and reveal a simple but profound underlying connection between DDIM and SDS. Following this insight, we propose FSD to tackle the diversity degradation challenge. By using a consistent noise sampling schedule that aligns noise coarsely in 3D space, FSD breaks free from the maximum-likelihood-seeking nature of SDS and could generate diverse results without compromising generation quality.

Limitations and future works Although FSD could improve the diversity of 3D generation, we found that it is still difficult to generate 3D models as diverse as the images generated through DDIM generation process. Moreover, we only found plausible theorems for FSD-guided 2D generation, and generalized FSD to 3D generation in an intuitive way. As we do not specify the noise function $\boldsymbol{\epsilon}(\boldsymbol{c})$ in the general form of FSD (Eq. (15)), we design $\boldsymbol{\epsilon}(\boldsymbol{c})$ manually to coarsely align noise prior in 3D space in this work, where better designs of $\boldsymbol{\epsilon}(\boldsymbol{c})$ may exist. We leave this for future work.

References

1. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning representations and generative models for 3d point clouds. In: International conference on machine learning. pp. 40–49. PMLR (2018) 1
2. Armandpour, M., Zheng, H., Sadeghian, A., Sadeghian, A., Zhou, M.: Re-imagine the negative prompt algorithm: Transform 2d diffusion into 3d, alleviate janus problem and beyond. arXiv preprint arXiv:2304.04968 (2023) 3, 6
3. Chang, P., Tang, J., Gross, M., Azevedo, V.C.: How i warped your noise: a temporally-correlated noise prior for diffusion models. In: The Twelfth International Conference on Learning Representations (2024), https://openreview.net/forum?id=pzElnMrgSD 11, 31
4. Changpinyo, S., Sharma, P., Ding, N., Soricut, R.: Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3558–3568 (2021) 5
5. Chen, R., Chen, Y., Jiao, N., Jia, K.: Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. arXiv preprint arXiv:2303.13873 (2023) 6
6. Deitke, M., Schwenk, D., Salvador, J., Weihs, L., Michel, O., VanderBilt, E., Schmidt, L., Ehsani, K., Kembhavi, A., Farhadi, A.: Objaverse: A universe of annotated 3d objects. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13142–13153 (2023) 1
7. Ge, S., Nah, S., Liu, G., Poon, T., Tao, A., Catanzaro, B., Jacobs, D., Huang, J.B., Liu, M.Y., Balaji, Y.: Preserve your own correlation: A noise prior for video diffusion models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 22930–22941 (2023) 11, 31
8. Gu, J., Zhai, S., Zhang, Y., Liu, L., Susskind, J.M.: Boot: Data-free distillation of denoising diffusion models with bootstrapping. In: ICML 2023 Workshop on Structured Probabilistic Inference {\&} Generative Modeling (2023) 12, 20
9. Guo, Y.C., Liu, Y.T., Shao, R., Laforte, C., Voleti, V., Luo, G., Chen, C.H., Zou, Z.X., Wang, C., Cao, Y.P., Zhang, S.H.: threestudio: A unified framework for 3d content generation. https://github.com/threestudio-project/threestudio (2023) 12, 25
10. Hessel, J., Holtzman, A., Forbes, M., Bras, R.L., Choi, Y.: Clipscore: A reference-free evaluation metric for image captioning. arXiv preprint arXiv:2104.08718 (2021) 19
11. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017) 20

12. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in neural information processing systems **33**, 6840–6851 (2020) 5
13. Ho, J., Salimans, T.: Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598 (2022) 3, 20, 21, 25, 26, 27
14. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: Lora: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685 (2021) 27
15. Huang, Y., Wang, J., Shi, Y., Qi, X., Zha, Z.J., Zhang, L.: Dreamtime: An improved optimization strategy for text-to-3d content creation. arXiv preprint arXiv:2306.12422 (2023) 3, 6, 8, 12
16. Karras, T., Aittala, M., Aila, T., Laine, S.: Elucidating the design space of diffusion-based generative models. Advances in Neural Information Processing Systems **35**, 26565–26577 (2022) 5, 28
17. Katzir, O., Patashnik, O., Cohen-Or, D., Lischinski, D.: Noise-free score distillation. arXiv preprint arXiv:2310.17590 (2023) 3, 6, 7, 26, 27
18. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (2023) 2
19. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014) 10, 26, 29, 30, 32
20. Laine, S., Hellsten, J., Karras, T., Seol, Y., Lehtinen, J., Aila, T.: Modular primitives for high-performance differentiable rendering. ACM Transactions on Graphics **39**(6) (2020) 2
21. Liang, Y., Yang, X., Lin, J., Li, H., Xu, X., Chen, Y.: Luciddreamer: Towards high-fidelity text-to-3d generation via interval score matching. arXiv preprint arXiv:2311.11284 (2023) 3, 4, 6, 9
22. Lin, C.H., Gao, J., Tang, L., Takikawa, T., Zeng, X., Huang, X., Kreis, K., Fidler, S., Liu, M.Y., Lin, T.Y.: Magic3d: High-resolution text-to-3d content creation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 300–309 (2023) 6
23. Liu, R., Wu, R., Van Hoorick, B., Tokmakov, P., Zakharov, S., Vondrick, C.: Zero-1-to-3: Zero-shot one image to 3d object. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9298–9309 (2023) 6
24. Liu, Y., Lin, C., Zeng, Z., Long, X., Liu, L., Komura, T., Wang, W.: Syncdreamer: Generating multiview-consistent images from a single-view image. arXiv preprint arXiv:2309.03453 (2023) 6, 20
25. Long, X., Guo, Y.C., Lin, C., Liu, Y., Dou, Z., Liu, L., Ma, Y., Zhang, S.H., Habermann, M., Theobalt, C., et al.: Wonder3d: Single image to 3d using cross-domain diffusion. arXiv preprint arXiv:2310.15008 (2023) 6
26. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. Advances in Neural Information Processing Systems **35**, 5775–5787 (2022) 4, 5, 28
27. Luo, S., Hu, W.: Diffusion probabilistic models for 3d point cloud generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2837–2845 (2021) 1
28. Ma, B., Deng, H., Zhou, J., Liu, Y.S., Huang, T., Wang, X.: Geodream: Disentangling 2d and geometric priors for high-fidelity and consistent 3d generation. arXiv preprint arXiv:2311.17971 (2023) 6
29. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM **65**(1), 99–106 (2021) 2

30. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Transactions on Graphics (ToG) **41**(4), 1–15 (2022) 2, 26
31. Oord, A., Li, Y., Babuschkin, I., Simonyan, K., Vinyals, O., Kavukcuoglu, K., Driessche, G., Lockhart, E., Cobo, L., Stimberg, F., et al.: Parallel wavenet: Fast high-fidelity speech synthesis. In: International conference on machine learning. pp. 3918–3926. PMLR (2018) 2, 6
32. Poole, B., Jain, A., Barron, J.T., Mildenhall, B.: Dreamfusion: Text-to-3d using 2d diffusion. arXiv preprint arXiv:2209.14988 (2022) 1, 3, 4, 6, 7, 9, 13, 19, 20, 21, 22, 23, 26, 27
33. Qiu, H., Xia, M., Zhang, Y., He, Y., Wang, X., Shan, Y., Liu, Z.: Freenoise: Tuning-free longer video diffusion via noise rescheduling. arXiv preprint arXiv:2310.15169 (2023) 11, 31
34. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021) 19
35. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. Advances in neural information processing systems **29** (2016) 19
36. Salimans, T., Ho, J.: Progressive distillation for fast sampling of diffusion models. arXiv preprint arXiv:2202.00512 (2022) 5
37. Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al.: Laion-5b: An open large-scale dataset for training next generation image-text models. Advances in Neural Information Processing Systems **35**, 25278–25294 (2022) 5
38. Sharma, P., Ding, N., Goodman, S., Soricut, R.: Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 2556–2565 (2018) 5
39. Shen, T., Gao, J., Yin, K., Liu, M.Y., Fidler, S.: Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. In: Advances in Neural Information Processing Systems (NeurIPS) (2021) 2
40. Shi, Y., Wang, P., Ye, J., Long, M., Li, K., Yang, X.: Mvdream: Multi-view diffusion for 3d generation. arXiv preprint arXiv:2308.16512 (2023) 6, 9, 12, 13, 19, 20, 23, 25
41. Smith, E.J., Meger, D.: Improved adversarial systems for 3d object generation and reconstruction. In: Conference on Robot Learning. pp. 87–96. PMLR (2017) 1
42. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. In: International conference on machine learning. pp. 2256–2265. PMLR (2015) 5
43. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. arXiv preprint arXiv:2010.02502 (2020) 3, 5, 7, 12, 13, 19, 20, 22, 26, 28, 30
44. Song, Y., Dhariwal, P., Chen, M., Sutskever, I.: Consistency models. arXiv preprint arXiv:2303.01469 (2023) 20
45. Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. arXiv preprint arXiv:2011.13456 (2020) 3, 4, 5, 9, 11, 22, 23, 27
46. Tang, B., Wang, J., Wu, Z., Zhang, L.: Stable score distillation for high-quality 3d generation. arXiv preprint arXiv:2312.09305 (2023) 3, 6

47. Wang, H., Du, X., Li, J., Yeh, R.A., Shakhnarovich, G.: Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12619–12629 (2023) 1, 3, 4, 6, 7, 11, 13, 19, 20, 21, 25, 26, 27
48. Wang, P., Xu, D., Fan, Z., Wang, D., Mohan, S., Iandola, F., Ranjan, R., Li, Y., Liu, Q., Wang, Z., et al.: Taming mode collapse in score distillation for text-to-3d generation. arXiv preprint arXiv:2401.00909 (2023) 3, 6, 27
49. Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., Wang, W.: Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. arXiv preprint arXiv:2106.10689 (2021) 2
50. Wang, Z., Lu, C., Wang, Y., Bao, F., Li, C., Su, H., Zhu, J.: Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. Advances in Neural Information Processing Systems **36** (2024) 3, 6, 7, 8, 12, 20, 21, 26, 27
51. Weng, H., Yang, T., Wang, J., Li, Y., Zhang, T., Chen, C., Zhang, L.: Consistent123: Improve consistency for one image to 3d object synthesis. arXiv preprint arXiv:2310.08092 (2023) 6
52. Wu, Z., Zhou, P., Yi, X., Yuan, X., Zhang, H.: Consistent3d: Towards consistent high-fidelity text-to-3d generation with deterministic sampling prior (2024) 12, 20
53. Xie, J., Zheng, Z., Gao, R., Wang, W., Zhu, S.C., Wu, Y.N.: Learning descriptor networks for 3d shape synthesis and analysis. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8629–8638 (2018) 1
54. Xie, X., Zhou, P., Li, H., Lin, Z., Yan, S.: Adan: Adaptive nesterov momentum algorithm for faster optimizing deep models. arXiv preprint arXiv:2208.06677 (2022) 10
55. Ye, J., Wang, P., Li, K., Shi, Y., Wang, H.: Consistent-1-to-3: Consistent image to 3d view synthesis via geometry-aware diffusion models. arXiv preprint arXiv:2310.03020 (2023) 6
56. Yu, X., Guo, Y.C., Li, Y., Liang, D., Zhang, S.H., Qi, X.: Text-to-3d with classifier score distillation. arXiv preprint arXiv:2310.19415 (2023) 3, 6
57. Zhang, S.H., Guo, Y.C., Gu, Q.W.: Sketch2model: View-aware 3d modeling from single free-hand sketches. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6012–6021 (2021) 1
58. Zhu, J., Zhuang, P.: Hifa: High-fidelity text-to-3d with advanced diffusion guidance. arXiv preprint arXiv:2305.18766 (2023) 3, 6, 8, 12, 25

Appendix

A Quantitative Results

We compute several metrics for 3D generation results with SDS [32,47] and FSD. The hyperparameters are kept the same as those in Appendix C. When using Stable Diffusion [43] as backbone, we use 16 prompts and 4 random seeds for each prompt (Tab. 1). When using MVDream [40] as backbone, we also use 16 prompts and 4 random seeds for each prompt (Tab. 2).

Table 1: Stable Diffusion [43] as backbone

Method	CLIP (↑)	IS (↑)	CROSS-FID (↑)
DDIM Images	33.72 ± 1.83	1.68 ± 0.55	-
SDS [32,47]	32.57 ± 1.43	1.58 ± 0.47	106.5 ± 58.3
FSD(ours)	$\mathbf{32.72 \pm 1.56}$	$\mathbf{1.78 \pm 0.49}$	$\mathbf{141.8 \pm 57.9}$

Table 2: MVDream [40] as backbone

Method	CLIP (↑)	IS (↑)	CROSS-FID (↑)
DDIM Images	34.64 ± 2.56	2.02 ± 0.47	-
SDS [32,47]	$\mathbf{30.93 \pm 3.40}$	1.77 ± 0.37	86.6 ± 33.4
FSD(ours)	30.12 ± 3.07	$\mathbf{2.13 \pm 0.35}$	$\mathbf{174.8 \pm 44.5}$

CLIP score We compute CLIP score [10,34] using ViT-B/32 to measure the semantic similarity between the renderings of the generated 3D object and the input text prompt. We sample 24 views for each prompt and each seed when computing CLIP score.

IS score We compute IS score [35] to measure both the image quality and diversity. We first compute the IS scores of sampled views for each prompt and then average the IS scores across prompts.

CROSS-FID score To directly measure the diversity of generation results, it is natural to measure the inception distance between different generated samples. We first sample 24 views for each prompt and each seed. Then we separate the images corresponding to random seeds 0, 1 and 2, 3 into two sets of images. We

compute FID [11] of the two sets of images and average the FID score across prompts. We term this score as CROSS-FID score since it is different from the standard way of using FID to evaluate GANs [11].

B Discussions

Difference with Consistent3D Recent work Consistent3D [52] also applied fixed noise when conducting SDS-like generation. In Consistent3D, they follow the idea of Consistent Training [44] and use the rendered image perturbed with fixed noise to approximate the starting point of the deterministic flow. In our method, for the same camera view, we also add fixed noise to the rendered image, but the noised image is used to simulate a variable in the middle of a PF-ODE trajectory, which is different from Consistent3D. Our FSD loss is also different from the CDS loss in Consistent3D, even when our view-dependent noise function gives the same noise for all camera views, implying an essential difference between our method and Consistent3D.

Connection to Signal-ODE Our reformulated ODE (Eq. (12)) is equivalent to the Signal-ODE presented in the concurrent and independent work BOOT [8], which aims to distill a fast image generator. When the diffusion model is changed to sample prediction in Eq. (12), our reformulated PF-ODE is the same as the Signal-ODE in BOOT. In BOOT, they let the student image generation model predict the clean variables $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ on the ODE trajectory, while our method uses images rendered from 3D representation θ to model the clean variables $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ on the ODE trajectory.

C Additional Experiment Results

C.1 Additional Generation Results

We provide additional 3D experiment results using FSD and other SDS-like methods in Fig. 9, Fig. 10 and Fig. 11.

Specifically, in Fig. 9, we present results of FSD and compare them to baseline methods, namely ProlificDreamer (VSD) [50], SDS [32, 47], CSD [13]. In Fig. 10, we provide additional comparisons between SDS and FSD with Stable Diffusion [43] as the backbone Diffusion Model. In Fig. 11, we provide additional comparisons between SDS and FSD with MVDream [40] as the backbone Diffusion Model. We do not provide experiment results with Diffusion Models designed for novel view synthesis (*e.g.* SyncDreamer [24]) since we found FSD does not improve the diversity of the back view of an object significantly given the front view when using Diffusion Models of such category.

Fig. 9: Comparisons between FSD and SDS-like baseline methods. We present results of FSD and compare them to baseline methods, namely ProlificDreamer (VSD) [50], SDS [32, 47], CSD [13]. We use random seeds from 0 to 3 for each prompt.

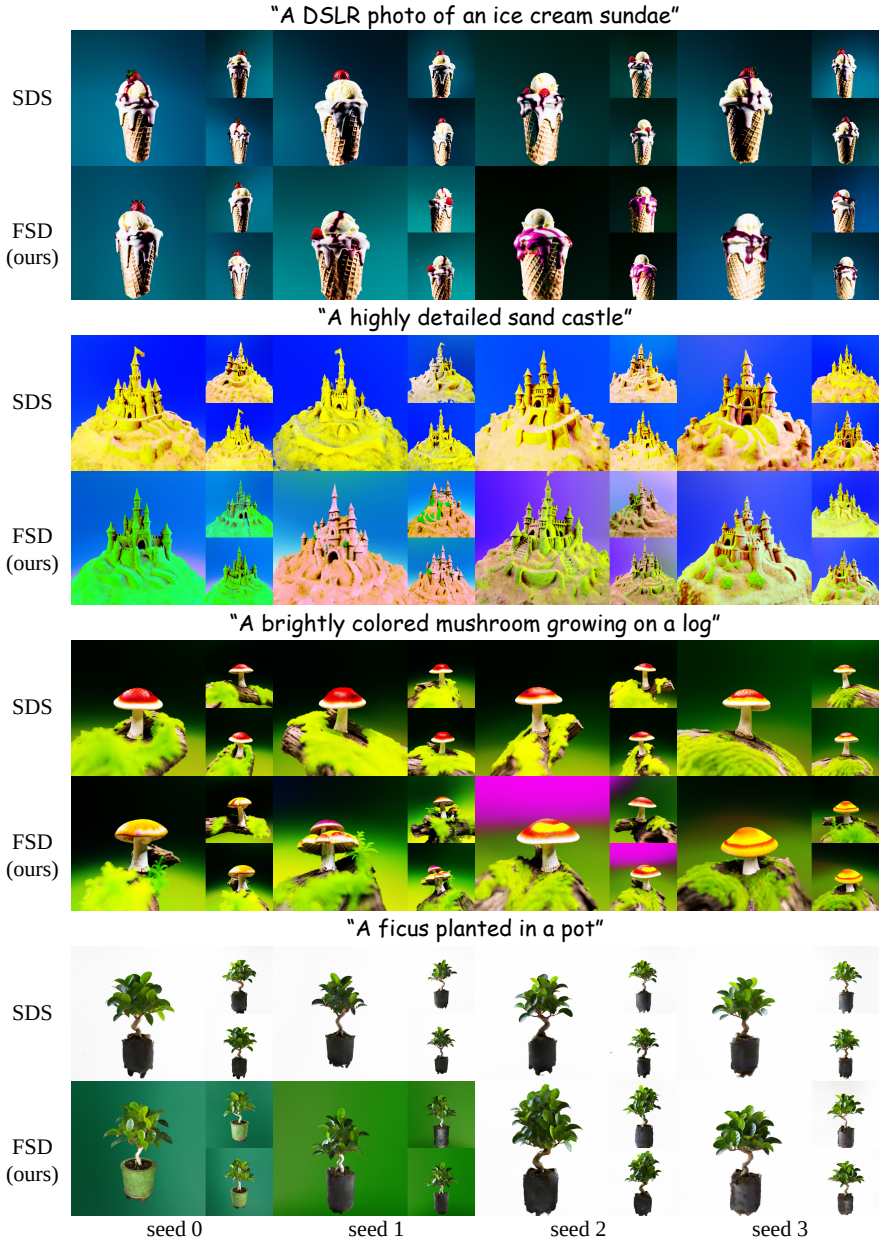


Fig. 10: Experiment results of FSD and SDS [32, 45] with Stable Diffusion [43] as backbone.

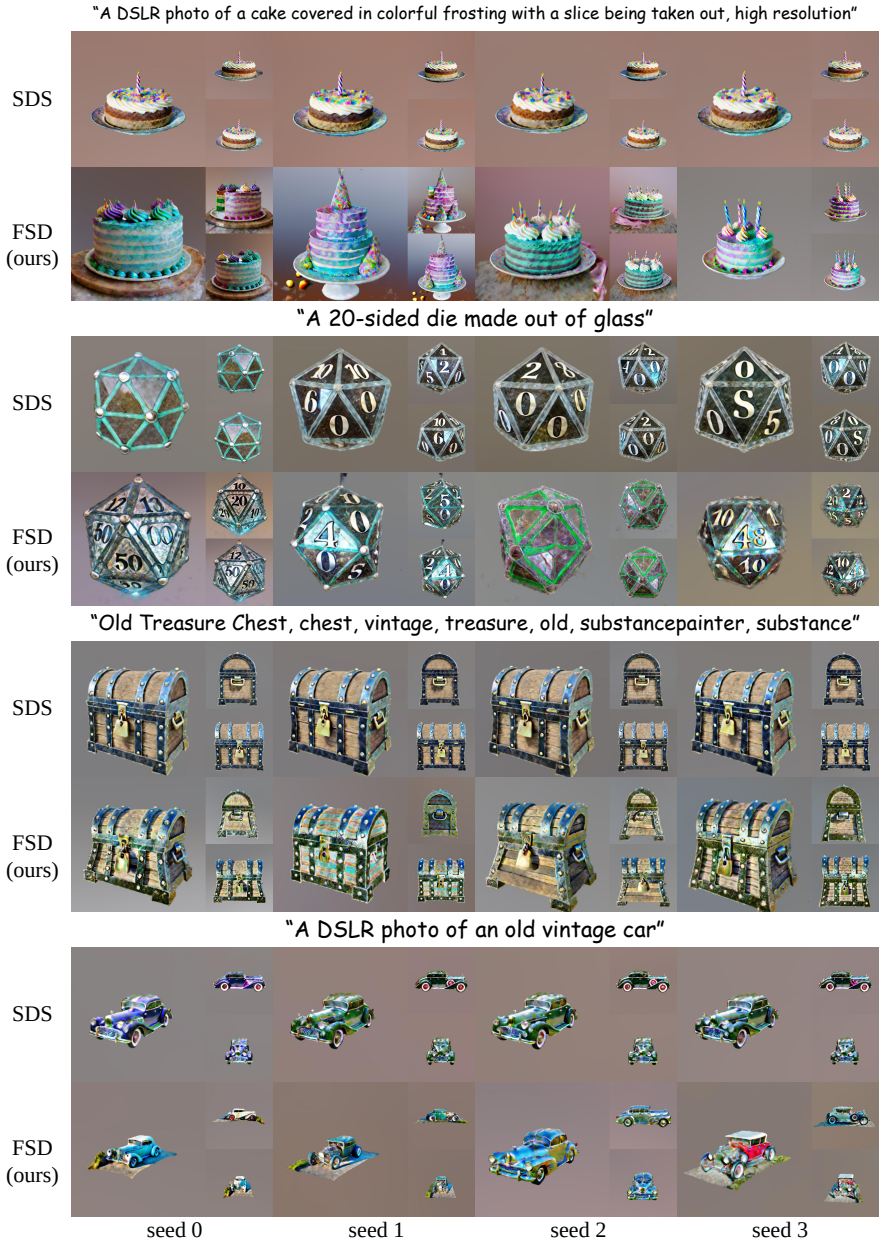


Fig. 11: Experiment results of FSD and SDS [32,45] with MVDream [40] as backbone.

Fig. 12: Ablation on other hyperparameters in $\epsilon(c)$. We use a parameter Θ to control the size of noise world map (H and W) in $\epsilon(c)$. When Θ is larger, the "radius r_+" (Eq. (29) and Eq. (30)) of the noise world map is smaller and FSD trends to generate smaller 3D models. In practice, we found FSD is prone to the parameter Θ.

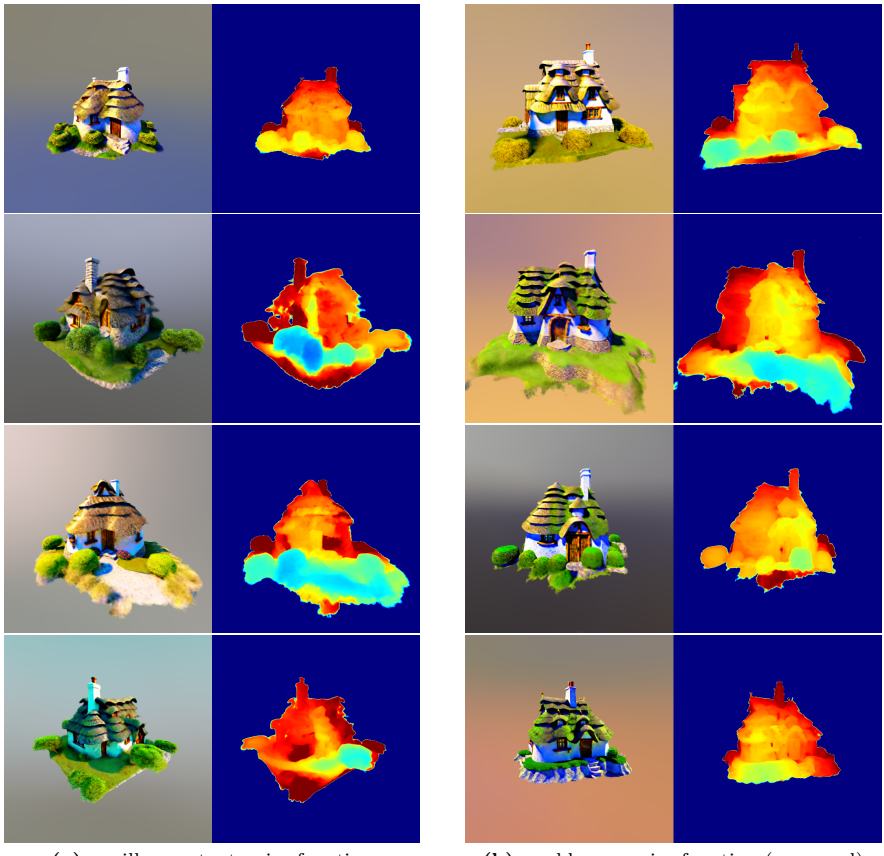
C.2 Additional Ablations

We use a parameter Θ to control the noise world-map's size (H and W). H and W are determined by Θ according to $H = H_{\text{hidden}} \frac{\pi}{\Theta}$ and $W = W_{\text{hidden}} \frac{2\pi}{\Theta}$. We visualize the results corresponding to different Θs in Fig. 12.

We also provide an additional comparison between the vanilla constant noise function and the world-map noise function in Fig. 13. We find the constant noise function may harm the geometry of the generation results.

C.3 Additional Experiments on Noise Prior

We provided additional generation results of FSD on 2D image space with shuffled or flipped initial noise in Fig. 14. The patches framed with the same color in the same row share the same initial noise patches $\tilde{\epsilon}$.



(a) vanilla constant noise function **(b)** world-map noise function (proposed)

Fig. 13: Ablation on design of $\epsilon(c)$. We use noise function $\epsilon(c)$ to align noise prior in FSD. We present RGB maps and depth maps of the generated samples using random seeds from 0 to 3. A vanilla constant function may result in holes on the surfaces. In contrast, the world-map noise function will not lead to any obvious flaws.

D Implementation Details

D.1 3D Experiments using FSD

The backbone Diffusion Model for Fig. 1 in the main text is MVDream [40]. The configuration for this figure is the same as Fig. 7 in the main text.

We use sqrt-annealing proposed by HiFA [58] for 3D experiments in this work. We use the same CFG [13] scale for SDS [40,47] and FSD. We use CFG=1 when applying image loss trick [58] since we find the reconstructed images are unrealistic when the CFG scale is large, which is not good guidance. We follow the default setting of threestudio [9] code base for other hyperparameters. We keep the iteration number and learning rate of FSD the same as SDS [40,47] in 3D

Fig. 14: Impact of initial noise $\tilde{\epsilon}$. We provide additional 2D experiment results generated by FSD show the impact of the initial noise $\tilde{\epsilon}$. The patches framed with the same color in the same row share the same initial noise patches.

experiments. We mainly conduct our experiments using NeRF representation [30] since SDS-like methods are not sensitive to the form of 3D representations.

D.2 2D Experiments using FSD

Here we describe the details of the experiments on 2D images with FSD in the main text. For Fig. 3 in the main text, we show the implementation details in Tab. 3. Below, we provide the loss functions for convenient reference.

Table 3: Implementation details on 2D experiments with FSD.

Methods Name	SDS [32, 47]	NFSD [17]	VSD [50]	FSD(ours)	DDIM [43]
Iteration Num	500	500	500	500	50
CFG [13]	100	7.5	7.5	7.5	7.5
Learning Rate	2e-2	2e-2	2e-2	3e-3	-
Optimizer	Adam [19]	Adam	Adam	Adam	-
Timestep Annealing	linear	linear	linear	linear	-

Let denote the prediction of Diffusion Models as $\boldsymbol{\epsilon}_y^t = \boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t|y,t)$ and c the classifier-free-guidance (CFG) [13] scale. Then

$$\nabla_\theta L_{\text{SDS}}^\theta = \mathbb{E}_{\boldsymbol{\epsilon},\boldsymbol{c},t}\left[\left(c\cdot(\boldsymbol{\epsilon}_y^t - \boldsymbol{\epsilon}_\emptyset^t) + (\boldsymbol{\epsilon}_\emptyset^t - \boldsymbol{\epsilon})\right)\frac{\partial \boldsymbol{g}_\theta(\boldsymbol{c})}{\partial\theta}\right] \tag{17}$$

is the loss function of SDS [32, 47], where $\emptyset$ is the empty prompt.

$$\nabla_\theta L_{\text{NFSD}}^\theta = \mathbb{E}_{\boldsymbol{\epsilon},\boldsymbol{c},t}\left[\left(c\cdot(\boldsymbol{\epsilon}_y^t - \boldsymbol{\epsilon}_\emptyset^t) + (\boldsymbol{\epsilon}_\emptyset^t - \boldsymbol{\epsilon}_{y_{\text{neg}}}^t)\right)\frac{\partial \boldsymbol{g}_\theta(\boldsymbol{c})}{\partial\theta}\right] \tag{18}$$

is the loss function of NFSD [17], where y_{neg} is a text negative prompt.

$$\nabla_\theta L_{\text{VSD}}^\theta = \mathbb{E}_{\boldsymbol{\epsilon},\boldsymbol{c},t}\left[\left(c\cdot(\boldsymbol{\epsilon}_y^t - \boldsymbol{\epsilon}_\emptyset^t) + (\boldsymbol{\epsilon}_\emptyset^t - \boldsymbol{\epsilon}_{\text{lora}}^t)\right)\frac{\partial \boldsymbol{g}_\theta(\boldsymbol{c})}{\partial\theta}\right] \tag{19}$$

is the loss function of VSD [50], where $\boldsymbol{\epsilon}_{\text{lora}}^t$ the Diffusion Model fine-tuned by LoRA [14].

$$\nabla_\theta L_{\text{FSD}}^\theta = \mathbb{E}_{\boldsymbol{c},t}\left[\left(c\cdot(\boldsymbol{\epsilon}_y^t - \boldsymbol{\epsilon}_\emptyset^t) + (\boldsymbol{\epsilon}_\emptyset^t - \boldsymbol{\epsilon}(\boldsymbol{c}))\right)\frac{\partial \boldsymbol{g}_\theta(\boldsymbol{c})}{\partial\theta}\right] \tag{20}$$

is the loss function of FSD. Additionally, $\boldsymbol{x}_t = \alpha_t \boldsymbol{g}_\theta(\boldsymbol{c}) + \sigma_t \boldsymbol{\epsilon}(\boldsymbol{c})$ for FSD.

Our re-implementation of L_{NFSD} is slightly different from the original [17] design of NFSD by ignoring the condition when $t < 200$, to keep the formulation simple. In this way, we find $\boldsymbol{\epsilon}_{\text{lora}}^t$ used in VSD [50] may work in a similar way as $\boldsymbol{\epsilon}_{y_{\text{neg}}}^t$ in NFSD [17]. As a result, single-particle VSD may not be able to generate diverse samples since it is still mode-seeking, aligning with the observation of ESD [48]. But we do not conduct further investigations which are out of the scope of our work.

E Theory of Flow Score Distillation

We will provide some additional preliminaries and proof of Proposition 1 in the main text in this section.

E.1 Diffusion PF-ODE

The Diffusion PF-ODE [45] can be written in the following form:

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = f(t)\boldsymbol{x}_t - \frac{1}{2}g^2(t)\nabla_{\boldsymbol{x}}\log p_t(\boldsymbol{x}_t|y), \quad \boldsymbol{x}_T \sim p_T(\boldsymbol{x}_T), \tag{21}$$

where $f(t) = \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}$, $g^2(t) = \frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\sigma_t^2$, according to DPM-solver [26]. To get Eq. (5) in the main text, we take the derivative of $\boldsymbol{x}_t/\alpha_t$:

$$\begin{aligned}
\frac{\mathrm{d}(\boldsymbol{x}_t/\alpha_t)}{\mathrm{d}t} =& \frac{1}{\alpha_t}\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} - \frac{\boldsymbol{x}_t}{\alpha_t^2}\frac{\mathrm{d}\alpha_t}{\mathrm{d}t} \\
=& \frac{1}{\alpha_t}\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} - \frac{\boldsymbol{x}_t}{\alpha_t}f(t) \\
=& -\frac{g^2(t)}{2\alpha_t}\nabla_{\boldsymbol{x}}\log p_t(\boldsymbol{x}_t|y) \\
=& -\frac{\frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\sigma_t^2}{2\alpha_t}\nabla_{\boldsymbol{x}}\log p_t(\boldsymbol{x}_t|y) \\
=& -(\frac{1}{\alpha_t}\frac{\mathrm{d}\sigma_t}{\mathrm{d}t} - \frac{\sigma_t}{\alpha_t^2}\frac{\mathrm{d}\alpha_t}{\mathrm{d}t})\sigma_t\nabla_{\boldsymbol{x}}\log p_t(\boldsymbol{x}_t|y) \\
=& \frac{\mathrm{d}(\sigma_t/\alpha_t)}{\mathrm{d}t}(-\sigma_t\nabla_{\boldsymbol{x}}\log p_t(\boldsymbol{x}_t|y)) \\
=& \frac{\mathrm{d}(\sigma_t/\alpha_t)}{\mathrm{d}t}\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t|t,y).
\end{aligned} \tag{22}$$

This equation is the scaled version of Diffusion PF-ODE under the notation of Karras *et al.* [16].

E.2 DDIM Sampling

We derive the DDIM [43] sampling algorithm in this section. According to the first order discretization of Eq. (22):

$$\frac{\boldsymbol{x}_t}{\alpha_t} - \frac{\boldsymbol{x}_s}{\alpha_s} = (\frac{\sigma_t}{\alpha_t} - \frac{\sigma_s}{\alpha_s})\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_s|s,y), \tag{23}$$

the sampling algorithm of DDIM [43] can be derived as the following equation:

$$\boldsymbol{x}_t = \alpha_t(\frac{\boldsymbol{x}_s - \sigma_s\boldsymbol{\epsilon}_\phi(x_s|y,s)}{\alpha_s}) + \sigma_t\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_s|y,s). \tag{24}$$

E.3 Proof of Proposition 1

We present a detailed derivation of Proposition 1 in the main text in this section. We define

$$\boldsymbol{x}_t = \alpha_t\hat{\boldsymbol{x}}_t^{\mathrm{c}} + \sigma_t\tilde{\boldsymbol{\epsilon}}, \tag{25}$$

for the reverse diffusion process and then we can apply change-of-variable on $\boldsymbol{x}_t$. Finally

$$\begin{aligned}
\frac{\mathrm{d}\hat{\boldsymbol{x}}_t^{\mathrm{c}}}{\mathrm{d}t} =& \frac{\mathrm{d}\frac{\boldsymbol{x}_t - \sigma_t\tilde{\boldsymbol{\epsilon}}}{\alpha_t}}{\mathrm{d}t} \\
=& \frac{\mathrm{d}(\boldsymbol{x}_t/\alpha_t)}{\mathrm{d}t} - \frac{\mathrm{d}(\sigma_t/\alpha_t)}{\mathrm{d}t}\tilde{\boldsymbol{\epsilon}} \\
=& \frac{\mathrm{d}(\sigma_t/\alpha_t)}{\mathrm{d}t}(\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t|t,y) - \tilde{\boldsymbol{\epsilon}}).
\end{aligned}$$

We also derive first-order discretization of FSD for 2D generation from Eq. (23):

$$\hat{\boldsymbol{x}}_t^{\mathrm{c}} - \hat{\boldsymbol{x}}_s^{o} = (\frac{\sigma_t}{\alpha_t} - \frac{\sigma_s}{\alpha_s})(\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_s|s, y) - \tilde{\boldsymbol{\epsilon}}), \tag{26}$$

F Visualize the Change-of-Variable

Even though we show FSD can generate similar images When using the same initial noise in the main text, there are notable differences between the generated images since FSD uses Adam [19] to update parameters while DDIM uses first-order discretization of Diffusion PF-ODE. We show generation results of FSD that use first-order discretization in Fig. 15.

We also visualize the "*change-of-variable*" trick we used in the derivation of the main theorem. We define our new variable: the *clean image* $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ according to the following equation:

$$\boldsymbol{x}_t = \alpha_t \hat{\boldsymbol{x}}_t^{\mathrm{c}} + \sigma_t \tilde{\boldsymbol{\epsilon}}. \tag{27}$$

Notably, there is also another similar but different concept: the *one-step estimated ground-truth image* $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$, defined by:

$$\boldsymbol{x}_t = \alpha_t \hat{\boldsymbol{x}}_t^{\mathrm{gt}} + \sigma_t \boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t|y, t). \tag{28}$$

We visualize the trajectory of $\boldsymbol{x}_t$, $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ and $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$ in the same DDIM generation process in Fig. 16. As $\hat{\boldsymbol{x}}_t^{\mathrm{gt}} - \hat{\boldsymbol{x}}_t^{\mathrm{c}} = \frac{\sigma_t}{\alpha_t}(\tilde{\boldsymbol{\epsilon}} - \boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t|y, t)) \propto \nabla_\theta L_{\mathrm{FSD}}^\theta$ implies, we can see DDIM as a process that tries to align *clean image* with the *one-step estimated ground-truth image* generated by the Diffusion Model. We also visualize $\boldsymbol{x}_t$, $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ and $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$ of FSD and SDS in Fig. 17 and Fig. 18, respecitively.

G Algorithm for Flow Score Distillation

We provide a summarized algorithm for Flow Score Distillation in Algorithm 1.

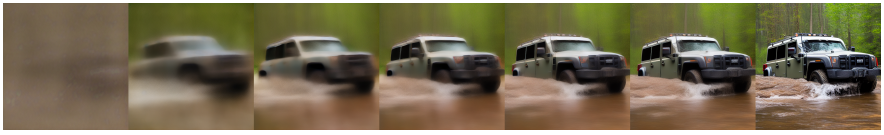
Algorithm 1 Flow Score Distillation

Input: Text-to-image Diffusion Model $\boldsymbol{\epsilon}_\phi$ and prompt y. Learning rate η for parameters of the 3D representation. A monotonically decreasing function $t(\tau)$. Blending factor β.
Compute $\boldsymbol{\epsilon}_b \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and $\boldsymbol{\epsilon}_p \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.
for $\tau \in [0, \tau_{\mathrm{end}}]$ **do**
  Randomly sample camera parameter $\boldsymbol{c}$.
  Render image $\boldsymbol{g}_\theta(\boldsymbol{c})$ and opacity mask $\boldsymbol{M}$ from 3D representation θ.
  Randomly sample $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.
  $\boldsymbol{\epsilon}(\boldsymbol{c}) \leftarrow \sqrt{\beta} \cdot \left((1 - \boldsymbol{M}) \odot \boldsymbol{\epsilon}_b + \boldsymbol{M} \odot \boldsymbol{W}_{(W\frac{\phi_{\mathrm{cam}}}{2\pi}, H\frac{\theta_{\mathrm{cam}}}{\pi})}(\boldsymbol{\epsilon}_p)\right) + \sqrt{1-\beta} \cdot \boldsymbol{\epsilon}$.
  $\theta \leftarrow \theta - \eta \cdot (\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t|y, t) - \boldsymbol{\epsilon}(\boldsymbol{c}))\frac{\partial \boldsymbol{g}_\theta(\boldsymbol{c})}{\partial \theta}$
end for

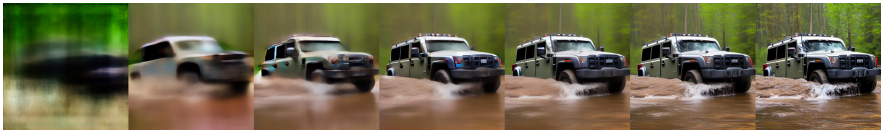
(a) FSD (first-order discretization) **(b)** DDIM [43]

Fig. 15: Generation results of FSD and DDIM. We apply FSD on 2D image generation using first-order discretization Eq. (26) instead of Adam [19]. In this case, we find FSD is the same as DDIM [43] except some negligible differences, which may come from the differences on handling initial conditions.

(a) *noisy image* $\boldsymbol{x}_t$



(b) *clean image* $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$



(c) *one-step estimated ground-truth image* $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$

Fig. 16: Visualization of $\boldsymbol{x}_t$, $\hat{\boldsymbol{x}}_t^{\mathbf{c}}$ and $\hat{\boldsymbol{x}}_t^{\mathbf{gt}}$ in DDIM generation process. $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ is different from $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$. Moreover, one can see DDIM generation process as aligning $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ with $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$ since $\hat{\boldsymbol{x}}_t^{\mathrm{gt}} - \hat{\boldsymbol{x}}_t^{\mathrm{c}} \propto \nabla_\theta L_{\mathrm{FSD}}^\theta$

H Practical Designing Rules for $\boldsymbol{\epsilon}(\boldsymbol{c})$

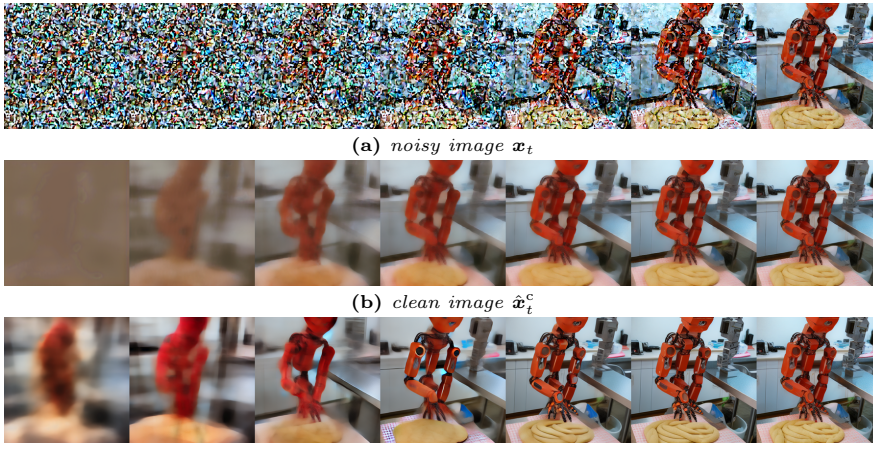
H.1 Practical Designing Rules

We do not specify the form of $\boldsymbol{\epsilon}(\boldsymbol{c})$ in the general form of FSD in the main text. However, it is intuitive to align $\boldsymbol{\epsilon}(\boldsymbol{c})$ in 3D space like the noise priors used in Video Diffusion Models [3, 7, 33]. We have tried several designs of $\boldsymbol{\epsilon}(\boldsymbol{c})$ and summarized several design rules for designing $\boldsymbol{\epsilon}(\boldsymbol{c})$ as well as the related potential problems if violating the design rules for $\boldsymbol{\epsilon}(\boldsymbol{c})$:

- The noise associated with each camera view conditioning on camera view should be an uncorrelated Gaussian noise. *i.e.* $\boldsymbol{\epsilon}(\boldsymbol{c})|\boldsymbol{c} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. If not so, the input image may be out-of-distribution for the pretrained text-to-image Diffusion Models.
- The noise associated with different camera views should be aligned in 3D space. Otherwise, FSD may degrade to the original SDS. This is consistent with the observation of recent work [7] on Video Generation, which showed that the noise maps corresponding to different video frames are highly correlated.
- The noise should not be only aligned in specific points in the 3D space. This may lead to broken geometry since the convergence speed in 3D space is not uniform (see analysis on failure of constant noise function in the main text).

H.2 World-map Noise Function

We will take a deeper analysis of the property of the world map noise function in the main text. We will discuss how our proposed design rules are followed.

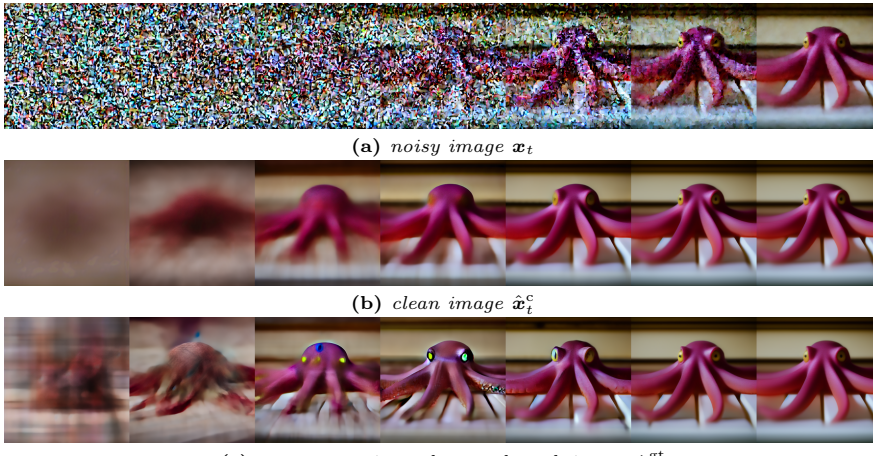


(a) *noisy image* $\boldsymbol{x}_t$

(b) *clean image* $\hat{\boldsymbol{x}}_t^{\mathbf{c}}$

(c) *one-step estimated ground-truth image* $\hat{\boldsymbol{x}}_t^{\mathbf{gt}}$

Fig. 17: Visualization of $\boldsymbol{x}_t$, $\hat{\boldsymbol{x}}_t^{\mathbf{c}}$ and $\hat{\boldsymbol{x}}_t^{\mathbf{gt}}$ in FSD generation process. Compared to DDIM, FSD uses Adam [19] to update parameters. As a result, FSD needs more iterations to converge (See Tab. 3).



(a) *noisy image* $\boldsymbol{x}_t$

(b) *clean image* $\hat{\boldsymbol{x}}_t^{\mathbf{c}}$

(c) *one-step estimated ground-truth image* $\hat{\boldsymbol{x}}_t^{\mathbf{gt}}$

Fig. 18: Visualization of $\boldsymbol{x}_t$, $\hat{\boldsymbol{x}}_t^{\mathbf{c}}$ and $\hat{\boldsymbol{x}}_t^{\mathbf{gt}}$ in SDS generation process. Compared to FSD, SDS adds random noise instead of fixed noise. As a result, the estimated GT images vary a lot and SDS needs a larger learning rate to converge (See Tab. 3).

As noted in the main text, the local texture of the added noise is highly correlated with the generated images. In 3D spaces, it is natural to identify points, that correspond to similar points on the noise maps when projected onto the image planes of different camera views $\boldsymbol{c}$, as the points of high convergence speed. Let us denote those points as $\boldsymbol{p}_+$. For simplicity, we study the $\boldsymbol{p}_+$ that corresponds to the center points in image space.

For object-centric generation, a common camera view sampling strategy is to sample camera positions at $(r_{\text{cam}}, \theta_{\text{cam}}, \phi_{\text{cam}})$ under spherical coordinate and force the camera to look at the center point. When we apply a constant noise function as $\boldsymbol{\epsilon}(\boldsymbol{c})$, the center point in 3D space is a $\boldsymbol{p}_+$ point. As a result, FSD trends to generate geometry with holes. When applying world-map noise function proposed in main text, $\boldsymbol{p}_+$ is located at $(r_+, \theta_{\text{cam}}, \phi_{\text{cam}})$. One can compute that

$$r_+^{\theta_{\text{cam}}} \approx \frac{2\tan\frac{\text{FOV}}{2}}{2\tan\frac{\text{FOV}}{2} + \Theta} \cdot r_{\text{cam}} \tag{29}$$

if consider nearby views with slightly different θ_{cam} and

$$r_+^{\phi_{\text{cam}}} \approx \frac{2\tan\frac{\text{FOV}}{2}}{2\tan\frac{\text{FOV}}{2} + \Theta \cdot \sin\theta_{\text{cam}}} \cdot r_{\text{cam}} \tag{30}$$

if consider views with slightly different ϕ_{cam}. In this way, we align nearby views coarsely. Moreover, for different camera parameters, r_+ is different, avoiding nonuniform convergence speed in 3D space.