Consistent Flow Distillation for Text-to-3D Generation

Runjie Yan
Electrical and Computer Engineering
UC San Diego

Yinbo Chen
Electrical and Computer Engineering
UC San Diego

Xiaolong Wang
Electrical and Computer Engineering
UC San Diego

Abstract

Score Distillation Sampling (SDS) has shown great progress in distilling image-generative models for 3D generation, while its maximum-likelihood-seeking behavior significantly degrades the visual quality. Recently, Flow Score Distillation (FSD) tackled this issue by using deterministic noise and annealing timesteps in SDS, and connected this process to Denoise Diffusion Implicit Model (DDIM) sampling for 2D images. However, its improvement in 3D generation quality is still limited. In this work, we propose Consistent Flow Distillation (CFD). We focus on the gradient-based sampling perspective and identify that a key in this process is to make the 2D image flow from different views consistent on the 3D object with correct correspondence. To achieve this, we develop a multi-view consistent Gaussian noise on the 3D object, which can be rendered from different views to compute the flow gradient. Our experiments demonstrate that with consistent flows, CFD significantly improves over previous methods in text-to-3D generation. Project page: `https://runjie-yan.github.io/cfd/`.

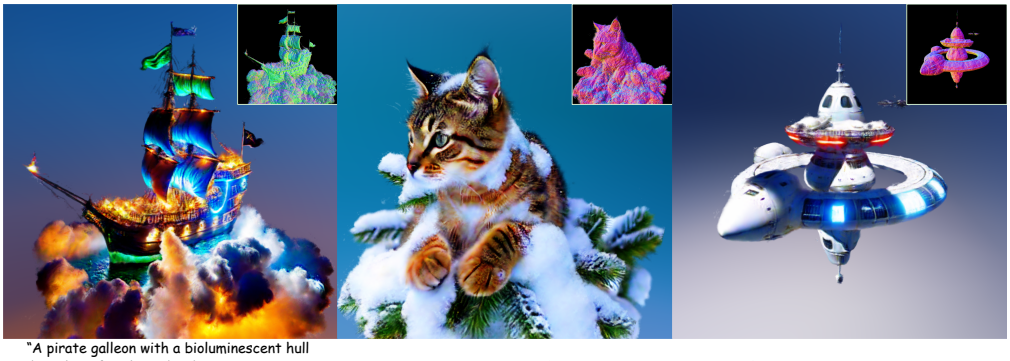
1 Introduction

3D content generation has gained increasing attention in recent years for its wide range of applications. However, it is expensive to create high-quality 3D assets or scan objects in the real world. The scarcity of 3D data has been a primary challenge in 3D generation. On the other hand, image synthesis has witnessed great progress, particularly with diffusion models trained on large-scale datasets with massive high-quality and diverse images. Leveraging the 2D generative knowledge for 3D generation by model distillation has become a research direction of key importance.

Score Distillation Sampling [30] (SDS) pioneered the paradigm. It uses a pretrained text-to-image diffusion model to optimize a single 3D representation such that the rendered views seek a maximum likelihood objective. Several subsequent efforts [1, 12, 14, 18, 39, 41, 52] have been made to improve SDS, while the maximum-likelihood-seeking behavior remains, which has a detrimental effect on the visual quality and diversity. Variational Score Distillation [44] (VSD) tackles this issue by treating the 3D representation as a random variable instead of a single point as in SDS. However, the random variable is simulated by particles in VSD. Single-particle VSD is theoretically equivalent to SDS [41], while the optimization-based sampling is k times slower with k particles.

Recently, Flow Score Distillation [48] (FSD) modified SDS with fixed added noise and annealing timesteps, and shows that for 2D image generation, when updating with first-order discretization instead of using optimizers like Adam [16], it is equivalent to a Denoise Diffusion Implicit Model [37] (DDIM) sampling process (PF-ODE [38]). Therefore, with these modifications, FSD for 2D image generation avoids seeking maximum likelihood, follows the correct distribution, and significantly

Preprint. Under review.



"A pirate galleon with a bioluminescent hull that glows faintly in the dark ocean waters, illuminating the ship's intricate carvings and sails as it silently navigates the waves"

"A cute cat covered by snow"

"A futuristic space station"

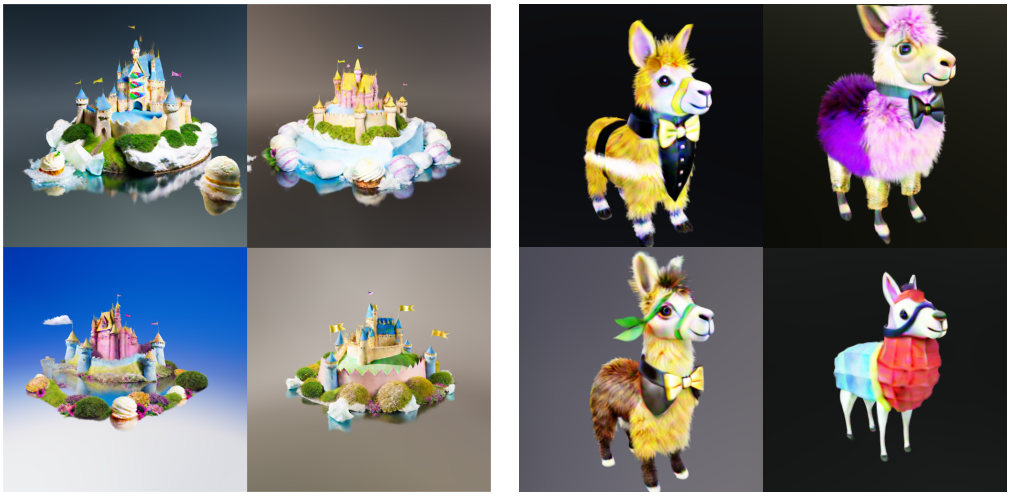
(a) NeRFs generated by CFD from scratch.

"An astronaut is riding a horse"

"A polar bear surfing a big wave"

"A steampunk owl with mechanical wings"

(b) 3D textured meshes generated by CFD from scratch.



"A Matte painting of a castle made of cheesecake surrounded by a moat made of ice cream"

"A llama in a tuxedo at a fancy gala"

(c) CFD can generate diverse and high-quality 3D samples from scratch by distilling different pretrained 2D diffusion models. Left: Distilling Stable Diffusion [37]; Right: Distilling MVDream [35].

Figure 1: **Text-to-3D samples of CFD.** CFD could generate diverse 3D samples by distilling text-to-image diffusion models. CFD could generate samples with single-stage pipeline (Fig. 1(c)), and could boost the quality of 3D samples with 2-stage pipeline (Fig. 1(a), Fig. 1(b)). See videos in our project page for additional generation results.

improves image quality. For the 3D generation, FSD adopts similar insights as in 2D and uses a deterministic view-dependent noise function, such that for a specific camera view, it always adds a fixed noise (for the foreground). Compared with SDS, they demonstrate more diverse results, but have limited improvement in visual quality.

In this work, we propose Consistent Flow Distillation (CFD), which distills a 3D representation by gradient-based sampling of consistent 2D image probability flow from different views in annealing timesteps. Following FSD, in the distillation process, different views are rendered from the 3D representation. The rendered image is viewed as the variable in the *clean flow* of image PF-ODE [38] trajectory defined by the pretrained image diffusion model, then gets updated by the gradient of the flow. With annealing timesteps in the flow, the rendered views gradually become realistic images and the 3D representation is sampled. In this process, we identify that the key is to apply consistent flows to the 3D representation. Intuitively, in 2D image generation, the same region is always associated with the same fixed noise for the correct flow sampling. Analogously, in 3D generation, the 2D image flows from different camera views should also use the noise patterns that are consistent on the object surface with correct correspondence. To achieve this, we design a *multi-view consistent Gaussian noise* based on Noise Transport Equation [4], which can compute the multi-view consistent noise with negligible cost. During the distillation process, the multi-view consistent Gaussian noise is rendered from different views to compute the gradient of 2D image flow.

We evaluate our method with different types of pre-trained 2D image diffusion models, and compare our results with state-of-the-art text-to-3D score distillation methods including SDS [30, 40], VSD [44] and FSD [48]. Both qualitative and quantitative experiments show the effectiveness of our approach compared with prior work. Our method generates 3D assets with realistic appearance and shape (Fig. 1(a), 1(b)) and can sample diverse 3D objects for the same text prompt with negligible extra computation cost compare with SDS (Fig. 1(c)).

In summary, our main contributions are:

- An in-depth discussion about using image diffusion PF-ODE and SDE to directly guide 3D generation. We present equivalent forms of the ODE and SDE so that random variables in them are clean images at any time in the diffusion process. We further identified that flow consistency is the key to unlocking the potential of the sampling perspective.
- A multi-view consistent Gaussian noise on the 3D object, that keeps pixel i.i.d. Gaussian property in a single view and has correct correspondence on object surface between views.
- A method to distill image diffusion models for 3D generation. It is as simple and efficient as SDS while having significantly better quality and diversity.

2 Preliminaries

2.1 Diffusion models and Probability Flow Ordinary Differential Equation (PF-ODE)

A forward diffusion process [9, 36] gradually adds noise to a data point $\boldsymbol{x}_0 \sim p_0(\boldsymbol{x}_0)$, such that the intermediate distribution $p_t(\boldsymbol{x}_t|\boldsymbol{x}_0)$ conditioned on initial sample $\boldsymbol{x}_0$ at diffusion timestep t is $\mathcal{N}(\alpha_t \boldsymbol{x}_0, \sigma_t^2 \boldsymbol{I})$, which can be equivalently write as

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

where $\alpha_0 = 1, \sigma_0 = 0$ at the beginning, and $\alpha_T \approx 0, \sigma_T \approx 1$ in the end, such that $p_T(\boldsymbol{x}_T)$ is approximately the standard Gaussian $\mathcal{N}(\mathbf{0}, \sigma_T^2 \boldsymbol{I})$. A diffusion model $\boldsymbol{\epsilon}_\phi$ is learned to reverse such process, typically with the following denoising training objective [9]:

$$\mathcal{L}_{\text{DM}}(\phi) = \mathbb{E}_{\boldsymbol{x}_0, \boldsymbol{\epsilon}, t}[w_t ||\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t) - \boldsymbol{\epsilon}||_2^2]. \tag{2}$$

After training, $\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t) \approx -\sigma_t \nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t)$, where $\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t)$ is termed *score function*.

A Probability Flow Ordinary Differential Equation [38] (PF-ODE) has the same marginal distribution as the forward diffusion process at any time t. The PF-ODE can be written as:

$$\frac{\mathrm{d}(\boldsymbol{x}_t/\alpha_t)}{\mathrm{d}t} = \frac{\mathrm{d}(\sigma_t/\alpha_t)}{\mathrm{d}t} \left(-\sigma_t \nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t)\right) \tag{3}$$

$$= \frac{\mathrm{d}(\sigma_t/\alpha_t)}{\mathrm{d}t} \boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t), \quad \boldsymbol{x}_T \sim p_T(\boldsymbol{x}_T). \tag{4}$$

A data point $\boldsymbol{x}_0$ can be sampled by starting from a Gaussian noise $\boldsymbol{x}_T \sim \mathcal{N}(\mathbf{0}, \sigma_T^2 \boldsymbol{I})$ and following the PF-ODE trajectory from $t = T$ to $t = 0$, typically with discretized timesteps and an ODE solver.

2.2 Differentiable 3D Representations

Differentiable 3D representations are typically parameterized by the learnable parameters θ and a differentiable rendering function $\boldsymbol{g}_\theta(\boldsymbol{c})$ to render images corresponding to the camera views $\boldsymbol{c}$. In many tasks, the gradient is first obtained on the rendered images $\boldsymbol{g}_\theta(\boldsymbol{c})$ and then backpropagated through the Jacobian matrix $\frac{\partial \boldsymbol{g}_\theta(\boldsymbol{c})}{\partial \theta}$ of the renderer to the learnable parameters θ.

Common 3D neural representations include Neural Radiance Field (NeRF) [3, 28, 29, 42, 47], 3D Gaussian Splatting (3DGS) [15], and Mesh [17, 34]. In this work, we perform experiments on various 3D representations and validate that our method is applicable for generation across a wide range of 3D representations.

3 Consistent Flow Distillation

We present Consistent Flow Distillation (CFD), which takes a pretrained and frozen text-to-image diffusion model and distills a 3D representation by the gradient from the probability flow of the 2D image diffusion model. We have an in-depth discussion about the ODE sampling perspective in FSD [48] and identify that a key in this process is to make the flow guidance consistent across different camera views (see Sec. 3.1). We further propose an SDE, a generalization of the clean flow ODE, that incorporates noise injection during optimization to enhance generation quality (see Sec. 3.2). To achieve the consistent flow, we propose an algorithm to compute a multi-view consistent Gaussian noise, which provides noise for different views with noise texture exactly aligned on the surface of the 3D object (see Sec. 3.3). Finally, we draw connections between CFD and other score distillation methods (see Sec. 3.4).

3.1 3D Generation with 2D Clean Flow Gradient

Given a pretrained text-to-image diffusion model $\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, y)$, let y denote the condition (text prompt), the conditional distribution $p(\boldsymbol{x}_0|y)$ can be sampled from the PF-ODE [38] trajectory from $t = T$ to $t = 0$, which takes the form

$$\mathrm{d}(\frac{\boldsymbol{x}_t}{\alpha_t}) = \underbrace{\mathrm{d}(\frac{\sigma_t}{\alpha_t})}_{-lr} \cdot \underbrace{\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, y)}_{\nabla \mathcal{L}}. \tag{5}$$

By following the diffusion PF-ODE, pure Gaussian noise is transformed to an image in the target distribution $p(\boldsymbol{x}_0|y)$. Thus PF-ODE can be viewed as guiding the refinement of a noisy image to a realistic image. Can we use image PF-ODE to directly guide the generation of a differentiable 3D representation θ through the refining process, with θ as its learnable parameters and $\boldsymbol{g}_\theta$ as its differentiable rendering function? A direct implementation can be substituting the noisy images in Eq. 5 with the rendered images $\boldsymbol{g}_\theta(\boldsymbol{c})$ at the camera view $\boldsymbol{c}$ by letting $\boldsymbol{x}_t = \alpha_t \boldsymbol{g}_\theta(\boldsymbol{c})$. By viewing $\mathrm{d}(\frac{\sigma_t}{\alpha_t})$ as the learning rate lr of an optimizer and $\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, y)$ as the loss gradient to $\frac{\boldsymbol{x}_t}{\alpha_t}$, the gradient can be backpropagated through the Jacobian matrix of the renderer $\boldsymbol{g}_\theta(\boldsymbol{c})$ to update θ according to

$$\Delta\theta = -lr \cdot \boldsymbol{\epsilon}_\phi(\alpha_t \boldsymbol{g}_\theta(\boldsymbol{c}), t, y) \frac{\partial \boldsymbol{g}_\theta(\boldsymbol{c})}{\partial \theta}. \tag{6}$$

However, such a direct attempt may not work (see Fig. 6 (a)), since the image $\boldsymbol{x}_t$ at diffusion timestep t contains Gaussian noise. It is hard for the images rendered by a 3D representation to match the noisy images $\frac{\boldsymbol{x}_t}{\alpha_t}$ in an image PF-ODE, particularly around the beginning $t = T$, when $\boldsymbol{x}_T$ is per-pixel independent Gaussian noise. It is generally impossible for a continuous 3D representation to be rendered as per-pixel independent Gaussian noise from all camera views simultaneously. As a result, the rendered views may be out-of-distribution (OOD) as the input to the pretrained image diffusion model, and therefore cannot get meaningful gradient as guidance.

A reformulated diffusion PF-ODE through *change-of-variable* is proposed in FSD [48]. For each trajectory $\{\boldsymbol{x}_t\}_{t\in[0,T]}$ of the *noisy variable* $\boldsymbol{x}_t$ in the original PF-ODE (Eq. 5), the new variable $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ is

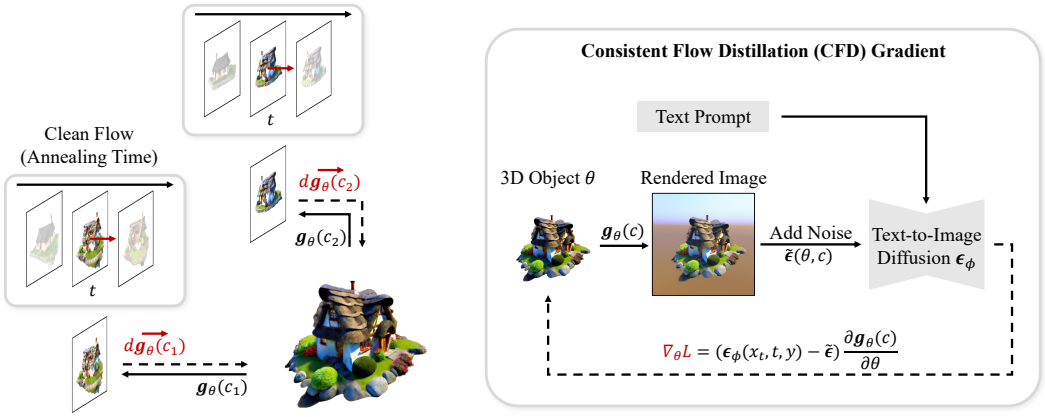


Figure 2: **Overview of CFD.** The 3D representation θ is generated with decreasing timesteps. At each timestep t, different views $g_\theta(c)$ are rendered. The 2D image clean flow provides the gradient at timestep t to the views and backpropagates to θ. The right shows the gradient computation: we add a multi-view consistent noise (see Fig. 3) to the rendered image and pass it into the frozen text-to-image diffusion model, gradient is calculated using the model prediction and then backpropagated to θ.

defined by

$$\hat{\boldsymbol{x}}_t^{\mathrm{c}} \triangleq \frac{\boldsymbol{x}_t - \sigma_t \tilde{\boldsymbol{\epsilon}}}{\alpha_t}, \tag{7}$$

where $\tilde{\boldsymbol{\epsilon}} = \frac{\boldsymbol{x}_T}{\sigma_T}$ is a constant for each ODE trajectory $\{\boldsymbol{x}_t\}_{t\in[0,T]}$ and is set as the initial noise. By Eq. 7 and Eq. 5, the new variable $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ follows the ODE [7, 48]

$$\mathrm{d}\hat{\boldsymbol{x}}_t^{\mathrm{c}} = \underbrace{\mathrm{d}(\frac{\sigma_t}{\alpha_t})}_{-lr} \cdot \underbrace{\left(\boldsymbol{\epsilon}_\phi(\alpha_t \hat{\boldsymbol{x}}_t^{\mathrm{c}} + \sigma_t \tilde{\boldsymbol{\epsilon}}, t, y) - \tilde{\boldsymbol{\epsilon}}\right)}_{\nabla \mathcal{L}}. \tag{8}$$

Changing the variable $\boldsymbol{x}_t$ of the original diffusion PF-ODE to the variable $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ makes directly using PF-ODE as a 3D guidance possible by providing the following properties: (i) $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ are clean images for all $t \in [0, T]$ (see Appx. Fig. 15), therefore, it can be substituted with the rendered clean images $\boldsymbol{g}_\theta(\boldsymbol{c})$. (ii) $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ is initialized from zero: $\hat{\boldsymbol{x}}_T^{\mathrm{c}} = \mathbf{0}$, which can be consistent with the 3D representation initialization (e.g. NeRF). (iii) The endpoint of the new ODE trajectory $\hat{\boldsymbol{x}}_0^{\mathrm{c}} = \boldsymbol{x}_0$ is a sample following the target distribution $p_0(\boldsymbol{x}_0)$. The new variable $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ is therefore termed *clean variable*. Note that $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ is different from the "*sample prediction*" of diffusion network for $\boldsymbol{x}_t$ (another non-noisy variable), which is not directly usable in this framework and we discuss for more details in Appx. Sec. G. We use *clean flow* to denote the ODE (Eq. 8) on the clean variable $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$.

In FSD, the 3D representation parameters θ is updated by clean flow gradient

$$\nabla_\theta \mathcal{L}_{\mathrm{FSD}}(\theta) = \mathbb{E}_c \left[\left(\boldsymbol{\epsilon}_\phi(\alpha_t \boldsymbol{g}_\theta(\boldsymbol{c}) + \sigma_t \tilde{\boldsymbol{\epsilon}}(\boldsymbol{c}), t, y) - \tilde{\boldsymbol{\epsilon}}(\boldsymbol{c})\right) \frac{\partial \boldsymbol{g}_\theta(\boldsymbol{c})}{\partial \theta} \right], \tag{9}$$

where $t = t(\tau)$ is a predefined monotonically decreasing timestep annealing function of the optimization time τ and $\tilde{\boldsymbol{\epsilon}}(\boldsymbol{c})$ is a deterministic noise function that only depends on the camera view, so that the noise term for a fixed view is always the same. Even though FSD improved on 2D image generation compared with prior score distillation methods [14, 40, 44], their quality improvement in 3D generation is yet limited and has over-smoothed texture like SDS [40]. We hypothesize this is because their noise function $\tilde{\boldsymbol{\epsilon}}(\boldsymbol{c})$ is suboptimal and does not generalize the sampling well from 2D to 3D. In 2D generation, a small local region on the image is always associated with the same noise pattern in clean flow ODE. However, in FSD for 3D generation, the same local region on the object surface might be associated with different noise patterns when computing gradient $\nabla_\theta \mathcal{L}_{\mathrm{FSD}}(\theta)$ from different views, since the noise function $\tilde{\boldsymbol{\epsilon}}(\boldsymbol{c})$ is aligned on a sphere which is independent of the object surface [48]. Adding different noise to the same local region of the 3D object makes the flow guidance inconsistent, and FSD may degenerate to having similar behaviors as SDS.

We identify that a key in the 3D sampling process is to make the 2D image flows consistent on the 3D object surface, which requires a multi-view consistent Gaussian noise function $\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})$ that is not only view-dependent but also depends on the object surface. We term the clean flows on a set of camera views with consistent noise $\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})$ as *consistent flow*. Then the gradient to update the 3D representation θ in our proposed method Consistent Flow Distillation (CFD) is

$$\nabla_\theta \mathcal{L}_{\text{CFD}}(\theta) = \mathbb{E}_c \left[\left(\boldsymbol{\epsilon}_\phi(\alpha_t \boldsymbol{g}_\theta(\boldsymbol{c}) + \sigma_t \tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c}), t, y) - \tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c}) \right) \frac{\partial \boldsymbol{g}_\theta(\boldsymbol{c})}{\partial \theta} \right], \tag{10}$$

where $t = t(\tau)$ is a predefined monotonically decreasing timestep annealing function and $\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})$ is the multi-view consistent Gaussian noise function (Sec. 3.3). The process is summarized in Fig. 2.

3.2 Guide 3D Generation with Diffusion SDE

Despite that PF-ODE and diffusion SDE can recover the same marginal distributions in theory, SDE based stochastic sampling may result in better generation quality as reported in prior works [13, 37, 38]. Inspired by this, we also propose to use image diffusion SDE to guide 3D generation.

To achieve this, we present a reverse-time SDE with a form similar to the clean flow ODE (Eq. 8):

$$\begin{cases} \mathrm{d}\hat{\boldsymbol{x}}_t^{\text{c}} = \underbrace{\left(\mathrm{d}(\frac{\sigma_t}{\alpha_t}) + \frac{\sigma_t}{\alpha_t} \beta_t \mathrm{d}t \right)}_{-lr} \cdot \underbrace{\left(\boldsymbol{\epsilon}_\phi(\alpha_t \hat{\boldsymbol{x}}_t^{\text{c}} + \sigma_t \tilde{\boldsymbol{\epsilon}}_t, t, y) - \tilde{\boldsymbol{\epsilon}}_t \right)}_{\nabla \mathcal{L}}, \\ \mathrm{d}\tilde{\boldsymbol{\epsilon}}_t = \tilde{\boldsymbol{\epsilon}}_t \beta_t \mathrm{d}t + \sqrt{2\beta_t} \mathrm{d}\bar{\boldsymbol{w}}_t, \end{cases} \tag{11}$$

with initial condition $\hat{\boldsymbol{x}}_T^{\text{c}} = \mathbf{0}$ and $\tilde{\boldsymbol{\epsilon}}_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, where $\bar{\boldsymbol{w}}_t$ is a standard Wiener process when time flows backwards from T to 0. We can further show that this SDE and its forward-time form are equivalent to the diffusion SDE presented by Song et al. [38] and EDM [13]. When we set $\beta_t = 0$, the SDE becomes deterministic and becomes the clean flow ODE. When $\beta_t \neq 0$, new Gaussian noise will be injected into $\tilde{\boldsymbol{\epsilon}}_t$ during the diffusion process, but $\tilde{\boldsymbol{\epsilon}}_t$ is still of unit variance throughout the whole process from T to 0. Furthermore, $\hat{\boldsymbol{x}}_t^{\text{c}}$ in this SDE still retains the "clean properties" of $\hat{\boldsymbol{x}}_t^{\text{c}}$ in the clean flow ODE. Thus, we also use clean flow to refer to this SDE. We provide detailed discussions and proofs on this SDE in Appx. F

The clean flow SDE implies a simple modification on Eq. 10 could make $\nabla_\theta \mathcal{L}_{\text{CFD}}(\theta)$ correspond to using SDE guidance. We only need to inject new Gaussian noise into $\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})$ during optimization by:

$$\tilde{\boldsymbol{\epsilon}}(\tau + 1) = \sqrt{1 - \gamma} \tilde{\boldsymbol{\epsilon}}(\tau) + \sqrt{\gamma} \boldsymbol{\epsilon}, \tag{12}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ is sampled at each optimization steps and γ is a predefined noise injection rate.

3.3 Multi-view Consistent Gaussian Noise $\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})$

To get consistent flow, a multi-view consistent Gaussian noise function $\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})$ is required, which (i) is a per-pixel independent Gaussian noise for all camera views $\boldsymbol{c}$; (ii) the noise patterns from different views have the correct correspondence according to the 3D object surface. It is non-trivial to satisfy all these properties with common warping and interpolation methods. The query rays from camera views $\boldsymbol{c}$ take continuous coordinates, simply using common interpolation methods such as bilinear may break the per-pixel independent property and result in bad generation quality (see Fig. 6 (b)).

Inspired by Integral Noise [4], we develop an algorithm that implements the multi-view consistent Gaussian noise with Noise Transport Equation. The Noise Transport Equation was originally proposed for warping between two frames in a video [4]. To use it in our 3D setting, we generalize the Noise Transport Equation to the warping between two different manifolds and compute the warping from different query camera views to the same reference space E_{ref}. As shown in Fig. 3, given a camera view $\boldsymbol{c}$, the query pixel $\boldsymbol{p}$ is first projected onto the surface of the object as camera-to-world $\boldsymbol{ctw}_{\boldsymbol{c}}(\boldsymbol{p})$ in the world space E_{world}, then we map those points from the surface to a reference space E_{ref} through a predefined mapping function $\mathcal{T}^{-1}$ (design details are in Appx. Sec. E). We define a high-resolution Gaussian noise map W on E_{ref}. Finally, we aggregate and return the noise value $G(\boldsymbol{p})$ for the query pixel $\boldsymbol{p}$ according to

$$G(\boldsymbol{p}) = \frac{1}{\sqrt{|\Omega_{\boldsymbol{p}}|}} \sum_{A_i \in \Omega_{\boldsymbol{p}}} W(A_i), \tag{13}$$

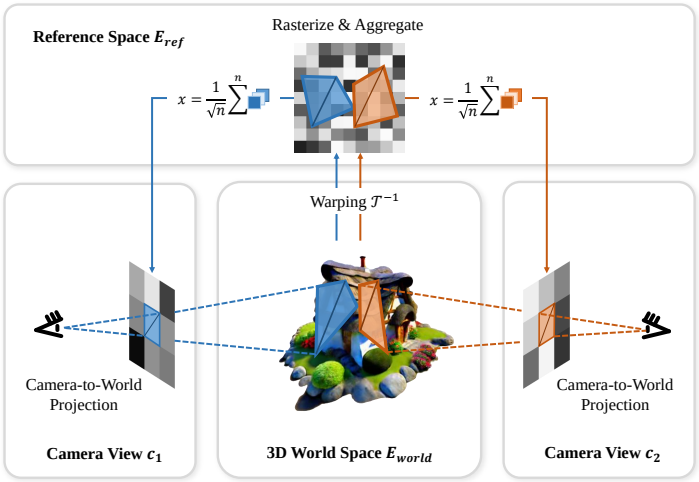


Figure 3: **Warping consistent noise for query views.** To obtain a query view noise map, for each pixel, its vertices are projected onto the object surface, then wrapped to the coordinates in a high-resolution noise map. The values within the region specified by the coordinates on the high-resolution noise map are summed up and normalized as the return pixel value in the query view noise.

where $\Omega_{\boldsymbol{p}} = \mathcal{T}^{-1}(\boldsymbol{ctw}_{\boldsymbol{c}}(\boldsymbol{p}))$ is the area covered by $\boldsymbol{p}$ after $\boldsymbol{p}$ being warped to E_{ref}, A_i is a noise cell in E_{ref}, and $W(A_i)$ is the noise value of unit variance at A_i. By first projecting query pixels from different camera views to the surface of the object in the world space E_{world}, two query pixels $\boldsymbol{p}_1, \boldsymbol{p}_2$ from two different camera views that look at the same region on the object will be projected to overlapped regions $\boldsymbol{ctw}_{\boldsymbol{c}_1}(\boldsymbol{p}_1)$, $\boldsymbol{ctw}_{\boldsymbol{c}_2}(\boldsymbol{p}_2)$ on the object. Warped by the same function $\mathcal{T}^{-1}$, they will cover overlapped regions $\Omega_{\boldsymbol{p}_1}, \Omega_{\boldsymbol{p}_2}$ and get correct correlation in noise map $G(\boldsymbol{p}_1), G(\boldsymbol{p}_2)$.

Our method can be also viewed as deriving a rendering function for the noisy variable $\boldsymbol{x}_t$ in the original form of PF-ODE (Eq. 5) by

$$\boldsymbol{x}_t(\theta, \boldsymbol{c}) = \alpha_t \boldsymbol{g}_\theta(\boldsymbol{c}) + \sigma_t \tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c}). \tag{14}$$

As discussed in Integral Noise [4], the warping of an image $\boldsymbol{g}_\theta(\boldsymbol{c})$ follows the transport equation that takes a similar form of Eq. 13, but with a different denominator $|\Omega_{\boldsymbol{p}}|$, instead of $\sqrt{|\Omega_{\boldsymbol{p}}|}$ for $\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})$, thus common 3D representation is incapable of rendering Gaussian Noise $\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})$, and it is needed to disentangle the noisy variable into the clean part $\boldsymbol{g}_\theta(\boldsymbol{c})$ and noisy part $\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})$. By disentanglement, we can handle the two parts that follow different rendering equations separately and achieve the rendering of the noisy variable for using image PF-ODE (and diffusion SDE) as a 3D guidance.

3.4 Comparison with SDS and FSD

SDS, FSD, and our CFD, all share a similar gradient form $\left(\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, y) - \tilde{\boldsymbol{\epsilon}}\right)\frac{\partial \boldsymbol{g}_\theta(\boldsymbol{c})}{\partial \theta}$ to update the 3D representation θ from a sampled rendered view. An overall comparison is summarized in Tab. 1. In SDS, t is typically randomly sampled from a range $[t_{\text{min}}, t_{\text{max}}]$, and $\tilde{\boldsymbol{\epsilon}}$ is a noise randomly sampled at each step. In contrast to SDS, FSD uses an annealing timestep $t(\tau)$ that decreases from t_{max} to t_{min}, the noise term $\tilde{\boldsymbol{\epsilon}}(c)$ is a function of camera view only, thus it is fixed from the same view and is not aware of the object surface. In our proposed CFD, the noise $\tilde{\boldsymbol{\epsilon}}(\theta, c)$ depends on both the object surface and the camera view, it is designed to let the noise from different views have correct correspondence according to the object surface. Notably, the object surface can slowly change during the generation process, so the noise for the same view in CFD is not strictly fixed even when $\gamma = 0$ in Eq. 12. We also report the gradient variance during training in Appx. Sec. H, where CFD has the lowest gradient variance compared with other methods.

Theoretically, when restricted to 2D image generation where $\boldsymbol{x} = \boldsymbol{g}_\theta(\boldsymbol{c})$, SDS is equivalent to seeking the maximum likelihood point in the noisy distribution p_t with a Gaussian distribution $\mathcal{N}(\alpha_t \boldsymbol{x}, \sigma_t^2 \boldsymbol{I})$ centered at the image $\boldsymbol{x}$. When the training objective of SDS is near optimal, their generation results are centered around a few modes [30]. In contrast, as proved in FSD [48], both FSD and our CFD are

	SDS [30, 40]	FSD [48]	CFD (ours, when $\gamma = 0$)
Timestep schedule	Random	Annealing	Annealing
Same view noise	Random	Fixed	Mostly fixed (surface-dependent)
Different views noise	Independent	Aligned on sphere	Aligned on object surface

Table 1: Comparision between SDS, FSD, and CFD.

sampling from the whole distribution p_0 and equivalent to a DDIM sampling process when using first-order discretization. Thus, both FSD and our CFD generate more diverse results. We also discuss the connection of our method to ISM [18] and SDI [25] in Appx. Sec. G.

4 Experiments

We show the generation results of CFD in Fig. 1(a), 1(b) and 1(c). To control for variables, we used the same codebase threestudio [8] to compare the our methods with various state-of-the-art baselines including SDS [30, 40], VSD [44] and FSD [48]. Since timestep annealing [12, 44, 52] has been proven to help improve generation quality [12, 44, 52], we also apply it to all baseline methods and CFD to study the effect of the noise function $\tilde{\epsilon}(\theta, c)$. For VSD [44], we use the original annealing schedule in their paper. We use the same linear annealing schedule for other methods. We set $\sqrt{1-\gamma} = 0.9999$ (in Eq. 12) for CFD and FSD. We mainly compare the performance of different score distillation methods by distilling two different pre-trained diffusion models, namely Stable Diffusion [37] and MVDream [35]. Stable Diffusion [37] is a well-known text-to-image diffusion model. MVDream [35] is a recent pose-aware text-to-multi-view-images diffusion model. Although the image quality produced by MVDream is inferior to that of Stable Diffusion, MVDream is generally able to generate consistent multi-view images, thereby mitigating the multi-face problems [1, 11, 30]. Refer to Appx. Sec. B for implementation details and Appx. Sec. D for details of experiment metrics.

4.1 Comparison with VSD, SDS and FSD

	3D-FID ↓	3D-IS ↑
SDS [30]	24.14	1.744 ± 0.419
VSD [44]	15.89	1.841 ± 0.282
FSD [48]	12.72	1.837 ± 0.264
CFD (ours)	**11.06**	$\mathbf{1.889 \pm 0.388}$

Table 2: 3D generation quality and diversity with Stable Diffusion [37]. 8 seeds for each of the 5 different prompts.

	3D-FID ↓	3D-IS ↑
SDS [30]	29.05	1.279 ± 0.089
FSD [48]	27.86	1.761 ± 0.233
CFD (ours)	**22.16**	$\mathbf{1.783 \pm 0.288}$

Table 3: 3D generation quality and diversity with MVDream [35]. 8 seeds for each of the 5 different prompts. VSD was not implemented with MVDream and thus not compared with in this setting.

We provide quantitative results in Tab. 2, 3, 4 and qualitative results in Fig. 4. As shown by quantitative and qualitative results, CFD outperforms all baseline methods and has better generation diversity and quality. We also provide addition results and comparison in Appx. Sec. A.

4.2 Ablation Study

Noise deterministic factor β. We follow FSD [48] to use a blending factor β to ablation the impact of noise function. Specifically, we add noise $\sqrt{\beta}\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c}) + \sqrt{1-\beta}\boldsymbol{\epsilon}$ to the rendered images in this experiment, where $\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})$ is the determined noise function, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ is the random noise sampled at each step of optimization. When $\beta = 0$, the noise is purely random and our method degenerates to SDS [30, 40]. When $\beta = 1$, the noise is deterministic and leverages the full power of CFD. As shown in Fig. 5, the generation performance generally increases as β increases. Note that new noise $\boldsymbol{\epsilon}$ will not be injected into $\tilde{\boldsymbol{\epsilon}}$ in this experiment, which is different from Eq. 12.

Ablation on the noise design and the flow space. As shown in Fig. 6: (a) When directly training θ with original PF-ODE using Eq. 6 with noisy variable, the training fails after several iterations. (b)

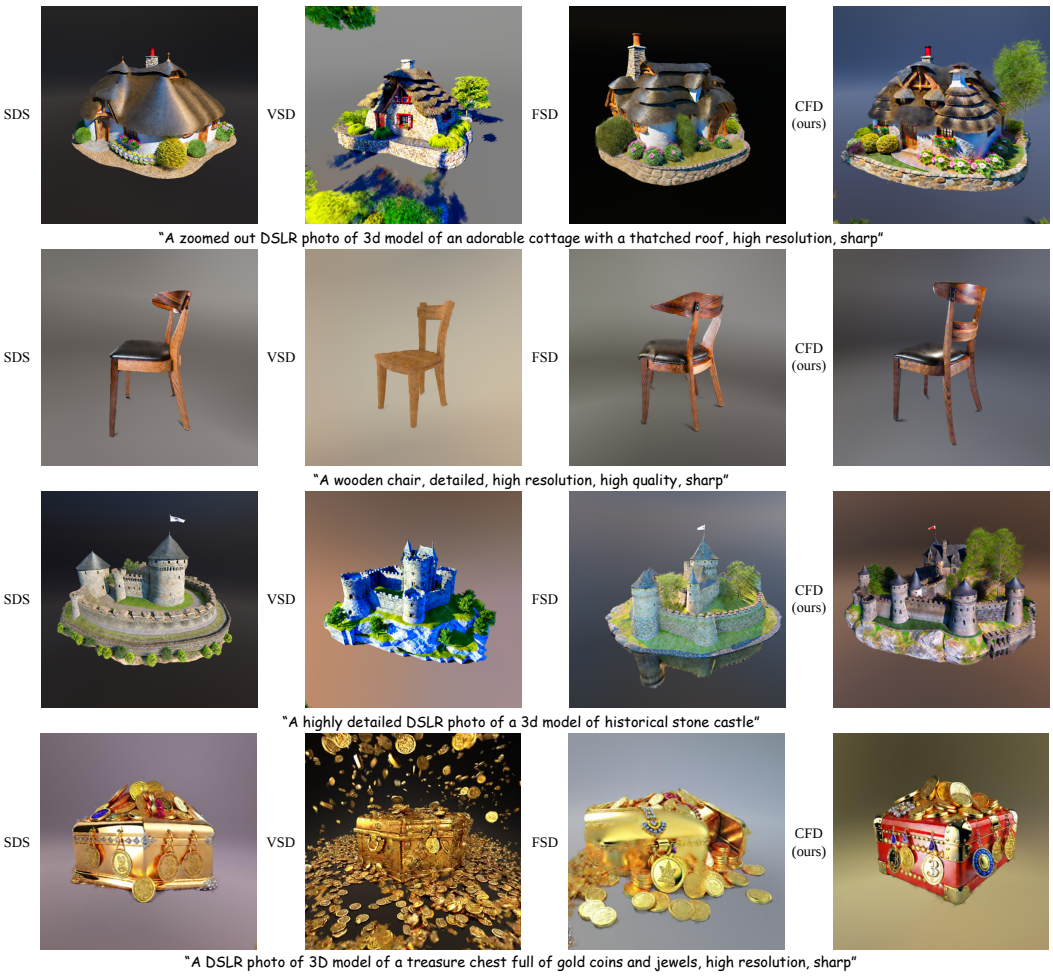


Figure 4: **Comparison with SDS, VSD and FSD.** We distill Stable Diffusion [37] in this experiment. We show 3 random samples for each of the methods in this figure. CFD outperforms SDS, VSD and FSD with better visual quality and diversity and has richer details.

	3D-CLIP ↑			
	B16	B32	L14	L14-336
SDS [30]	34.68±3.54	34.09±2.90	30.34±3.48	30.48±3.60
FSD [48]	34.85±3.00	34.32±2.85	31.06±3.02	31.11±2.84
CFD (ours)	**35.30±2.76**	**34.49±2.52**	**31.14±2.88**	**31.21±2.67**

Table 4: 3D generation quality with MVDream [35]. 1 seed for each of the 30 prompts.

Simply using bilinear interpolation instead of Noise Transport Equation leads to correlated pixel noise and generates bad results. (c) When using the noise function not aligned on the object surface as in FSD [48], the results are over-smoothed. (d) Our consistent flow distillation method with multi-view consistent Gaussian noise generates high-quality results. We also provide additional ablations on γ and design choices in Appx. Sec. C.

5 Related Work

Diffusion models Diffusion models [5, 9, 32, 33, 36, 38] are generative models that are learned to reverse a diffusion process. A diffusion process gradually adds noise to a data distribution, and the

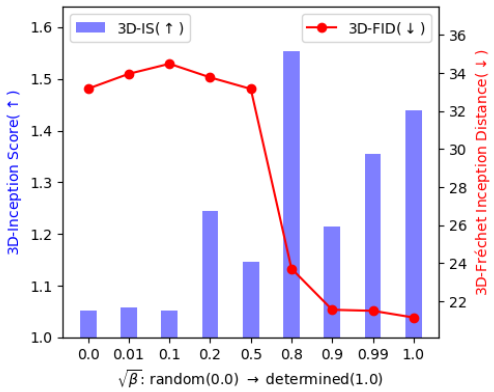
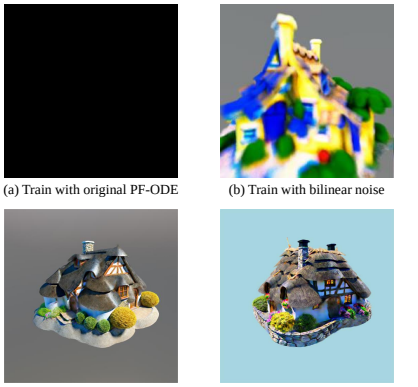


Figure 5: Ablation on β. Tested with one prompt and 4 different seeds for different βs.



(a) Train with original PF-ODE (b) Train with bilinear noise

(c) Train with world-map noise (FSD) (d) Train with consistent noise (CFD)

Figure 6: Ablation on the noise design and the flow space.

diffusion model is trained to reverse such an iterative process based on the score function. Denoise Diffusion Implicit Models (DDIM) [37] proposed a deterministic sampling method to speed up the sampling. Meanwhile, it is proved that a diffusion process corresponds to a Probability Flow Ordinary Differential Equation (PF-ODE) [38], which yields the same marginal distributions as the forward diffusion process at any timestep. Later works [13, 24, 31] demonstrate that DDIM can be viewed as the first-order discretization of the PF-ODE.

Score distillation sampling The score distillation sampling (SDS) paradigm for distilling 2D text-to-image diffusion models for 3D generation is proposed in DreamFusion [30] and SJC [40]. During the distillation process, the learnable 3D representation with differentiable rendering is optimized by the gradient to make the rendered view match the given text. Many recent works follow the SDS paradigm and studied for various aspects, including timestep annealing [12, 44, 52], coarse-to-fine training [6, 19, 44], analyzing the components [14], formulation refinement [1, 18, 39, 41, 44, 46, 48, 51, 52], geometry-texture disentanglement [6, 26, 44], addressing multi-face Janus problem replacing the text-to-image diffusion with novel view synthesis diffusion [21–23, 43, 45, 49] or multi-view diffusion [35].

6 Conclusion

In this paper, we proposed Consistent Flow Distillation. From a sampling perspective, we identified that using consistent flow to guide the 3D generation is the key to this process. We developed a multi-view consistent Gaussian noise with correct correspondence on the object surface and used it to implement the consistent flow. Our method can generate high-quality 3D representations by distilling 2D image diffusion models and shows improvement in quality and diversity compared with prior score distillation methods.

Limitations and broader impact. Although CFD can generate 3D assets of high fidelity and diversity, similar to prior works SDS, FSD, and VSD, the generation can take one to a few hours, and when distilling a text-to-image diffusion model, due to the properties of the teacher models, the distilled 3D representation sometimes may have multi-face Janus problem and may not be good for complex prompt. In practice, due to 3D representation flexibility and interference from other views, it is very hard to guarantee that the sampling process from a rendered view of the 3D object is exactly the same as sampling for 2D images given text. While our 3D consistent noise can reduce the interference and achieve better results, the flow for 3D rendered views may not be exactly the same as 2D flows of the initial noise. Also, like other generative models, it needs to pay attention to avoid generating fake and malicious content.

References

[1] 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)

[2] Bansal, A., Chu, H.M., Schwarzschild, A., Sengupta, S., Goldblum, M., Geiping, J., Goldstein, T.: Universal guidance for diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 843–852 (2023)

[3] Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields (2021)

[4] 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

[5] 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)

[6] Chen, R., Chen, Y., Jiao, N., Jia, K.: Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (October 2023)

[7] 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)

[8] 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)

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

[10] Ho, J., Salimans, T.: Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598 (2022)

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

[12] Huang, Y., Wang, J., Shi, Y., Tang, B., Qi, X., Zhang, L.: Dreamtime: An improved optimization strategy for diffusion-guided 3d generation. In: The Twelfth International Conference on Learning Representations (2024), https://openreview.net/forum?id=1bAUywYJTU

[13] 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)

[14] Katzir, O., Patashnik, O., Cohen-Or, D., Lischinski, D.: Noise-free score distillation. In: The Twelfth International Conference on Learning Representations (2024), https://openreview.net/forum?id=dlIMcmlAdk

[15] Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (2023)

[16] Kingma, D., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR). San Diega, CA, USA (2015)

[17] 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)

[18] 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)

[19] 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)

[20] Lin, S., Liu, B., Li, J., Yang, X.: Common diffusion noise schedules and sample steps are flawed. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. pp. 5404–5411 (2024)

[21] 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)

[22] Liu, Y., Lin, C., Zeng, Z., Long, X., Liu, L., Komura, T., Wang, W.: Syncdreamer: Generating multiview-consistent images from a single-view image. In: The Twelfth International Conference on Learning Representations (2024), https://openreview.net/forum?id=MN3yH2ovHb

[23] 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)

[24] 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)

[25] Lukoianov, A., Borde, H.S.d.O., Greenewald, K., Guizilini, V.C., Bagautdinov, T., Sitzmann, V., Solomon, J.: Score distillation via reparametrized ddim. arXiv preprint arXiv:2405.15891 (2024)

[26] 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)

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

[28] 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)

[29] 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)

[30] Poole, B., Jain, A., Barron, J.T., Mildenhall, B.: Dreamfusion: Text-to-3d using 2d diffusion. In: The Eleventh International Conference on Learning Representations (2023), https://openreview.net/forum?id=FjNys5c7VyY

[31] Salimans, T., Ho, J.: Progressive distillation for fast sampling of diffusion models. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=TIdIXIpzhoI

[32] 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)

[33] 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)

[34] 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)

[35] Shi, Y., Wang, P., Ye, J., Mai, L., Li, K., Yang, X.: MVDream: Multi-view diffusion for 3d generation. In: The Twelfth International Conference on Learning Representations (2024), https://openreview.net/forum?id=FUgrjq2pbB

[36] 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)

[37] Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=St1giarCHLP

[38] Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=PxTIG12RRHS

[39] Tang, B., Wang, J., Wu, Z., Zhang, L.: Stable score distillation for high-quality 3d generation. arXiv preprint arXiv:2312.09305 (2023)

[40] 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)

[41] 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)

[42] Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., Wang, W.: Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems. vol. 34, pp. 27171–27183. Curran Associates, Inc. (2021), https://proceedings.neurips.cc/paper_files/paper/2021/file/e41e164f7485ec4a28741a2d0ea41c74-Paper.pdf

[43] Wang, P., Shi, Y.: Imagedream: Image-prompt multi-view diffusion for 3d generation. arXiv preprint arXiv:2312.02201 (2023)

[44] 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)

[45] 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)

[46] Wu, Z., Zhou, P., Yi, X., Yuan, X., Zhang, H.: Consistent3d: Towards consistent high-fidelity text-to-3d generation with deterministic sampling prior. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9892–9902 (2024)

[47] Xu, Q., Xu, Z., Philip, J., Bi, S., Shu, Z., Sunkavalli, K., Neumann, U.: Point-nerf: Point-based neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5438–5448 (2022)

[48] Yan, R., Wu, K., Ma, K.: Flow score distillation for diverse text-to-3d generation (2024)

[49] 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)

[50] Yu, J., Wang, Y., Zhao, C., Ghanem, B., Zhang, J.: Freedom: Training-free energy-guided conditional diffusion model. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 23174–23184 (2023)

[51] Yu, X., Guo, Y.C., Li, Y., Liang, D., Zhang, S.H., QI, X.: Text-to-3d with classifier score distillation. In: The Twelfth International Conference on Learning Representations (2024), https://openreview.net/forum?id=ktG8Tun1Cy

[52] Zhu, J., Zhuang, P., Koyejo, S.: HIFA: High-fidelity text-to-3d generation with advanced diffusion guidance. In: The Twelfth International Conference on Learning Representations (2024), https://openreview.net/forum?id=IZMPWmcS3H

Appendix

A Additional Qualitative Comparison

We present more comparison between baseline methods and CFD in Fig. 7 and Fig. 8. We present additional generation results of CFD in Fig. 9, Fig. 13 and Fig. 11.

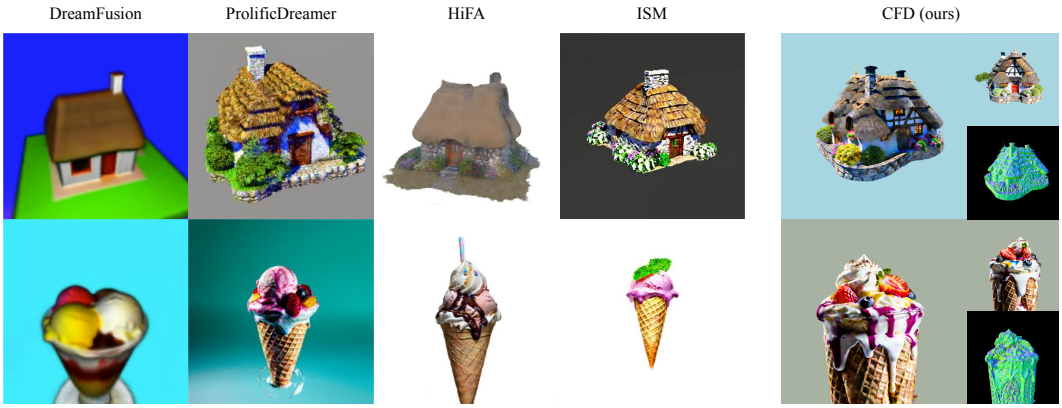


Figure 7: **Visual comparisons to baseline methods.** We compare render images of our method with baselines include DreamFusion [30], ProlificDreamer [44], HiFA [52], ISM [18]. The images of baselines are directly taken from their papers. Prompts: "A 3D model of an adorable cottage with a thatched roof"(top) and "A DSLR photo of an ice cream sundae"(bottom).

Figure 8: **Comparison with SDS.** We distill MVDream [35] in this experiment. We first generate coarse shape by distilling MVDream using SDS and CFD, then distill Stable Diffusion to refined the color with SDS and CFD, respectively. CFD outperforms SDS with better diversity and fidelity.

B Implementation Details

In this paper, we conduct experiments mainly on a single NVIDIA-GeForce-RTX-3090 or NVIDIA-L40 (when using soft shading). We use CFG [10] scale of 50 in experiments with SDS, FSD [48] and CFD (both MVDream [35] and Stable Diffusion [37]). We found CFD works the best with CFG scale of 50-100. We apply the same negative prompts [14, 27, 35] for different text prompts.

For simple prompts, we directly use CFD to distill Stable Diffusion (Fig. 4 and Fig. 9).

For mesh generation, we first use CFD to generate coarse shapes with MVDream [35]. Then we use CFD and follow the geometry and mesh refinement stages in VSD [44] with Stable Diffusion [37] to generate the mesh results in Fig. 1(b).

For complex prompts, we adopt a 2 stage pipeline (Fig. 1(a), Fig. 8, Fig. 13 and Fig. 11). We first generate coarse shape by distilling MVDream with CFD to avoid multi-face problem. Then we use CFD to distill Stable Diffusion to refine the details and colors (stage 2). We randomly replace the rendered image with rendered normal map with 0.1 probability to regularize the geometry in stage 2.

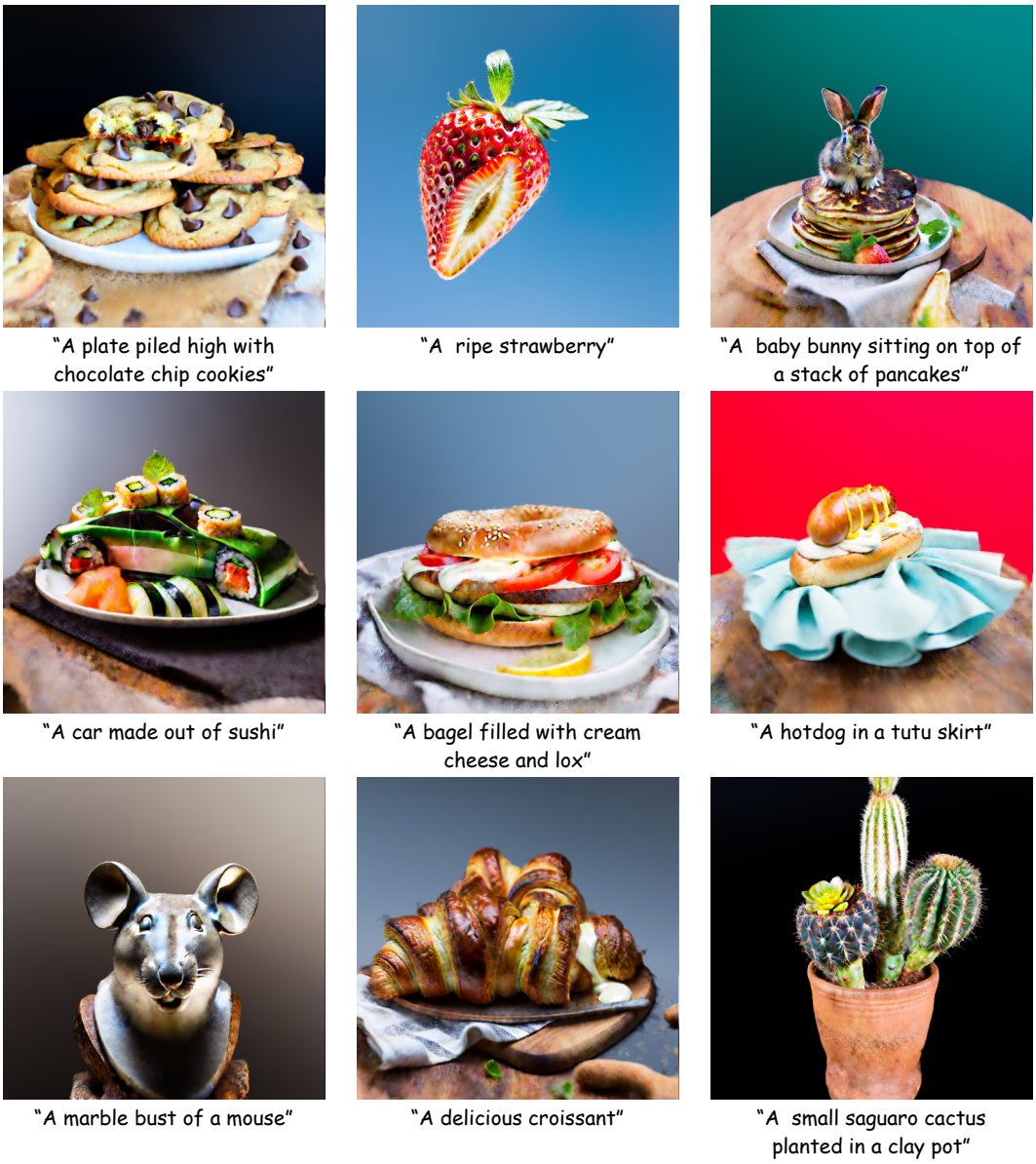


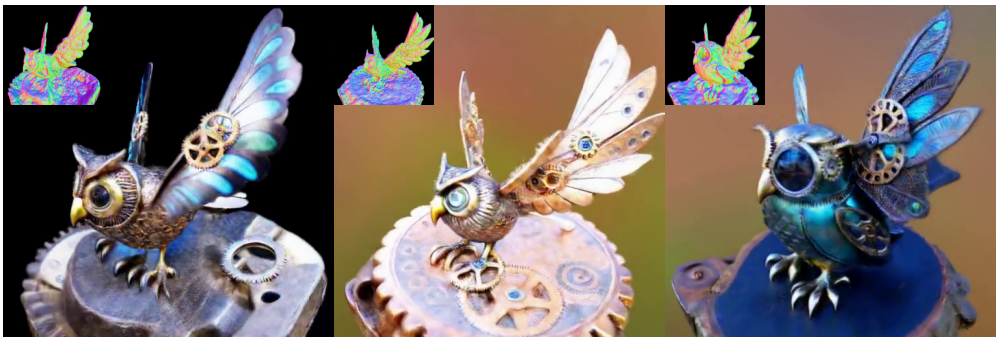
Figure 9: **NeRF results of CFD distilling Stable Diffusion [37].**

C Additional Ablations

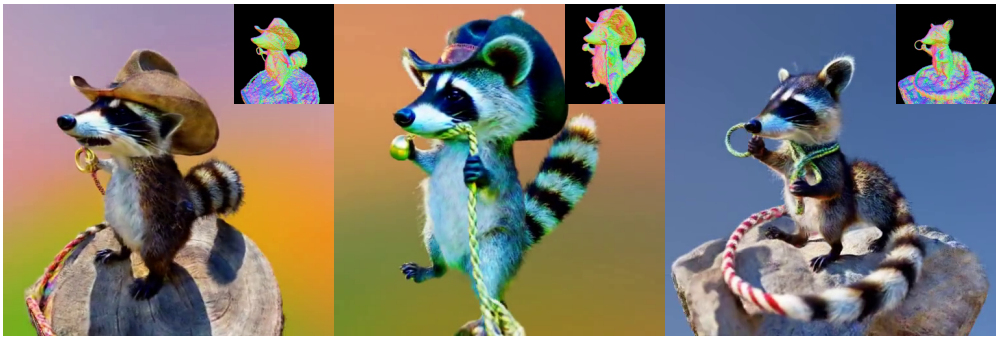
C.1 Ablation on Noise Injection Rate γ

Noise injection rate γ in Eq. 12 determines the rate at which new noise will be injected into the noise function. When $\gamma = 0$, no noise will be injected and CFD corresponds to using ODE guidance. When $\gamma > 0$, new noise will be injected, and $\tilde{\epsilon}(\theta, \boldsymbol{c})$ will gradually change. In this case, CFD corresponds to using SDE guidance.

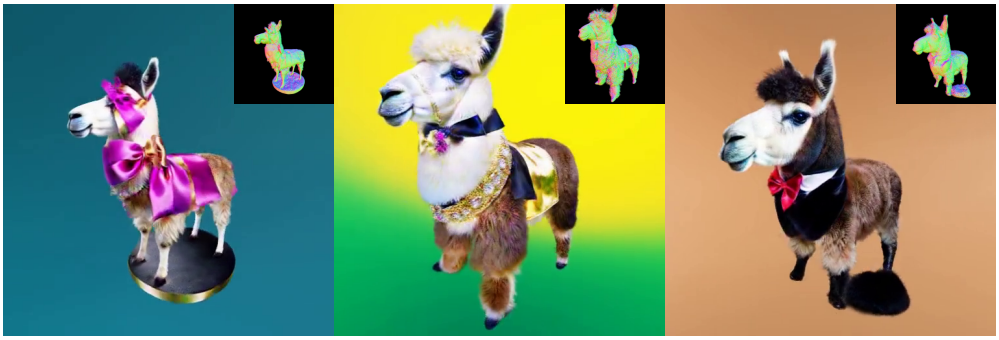
Using SDE based stochastic samplers may help to improve image generation quality as reported in prior works [13, 37, 38]. In Tab. 5, we also find that use a small nonzero γ seems to improve the performance of CFD, but the benefits seem to be marginal.



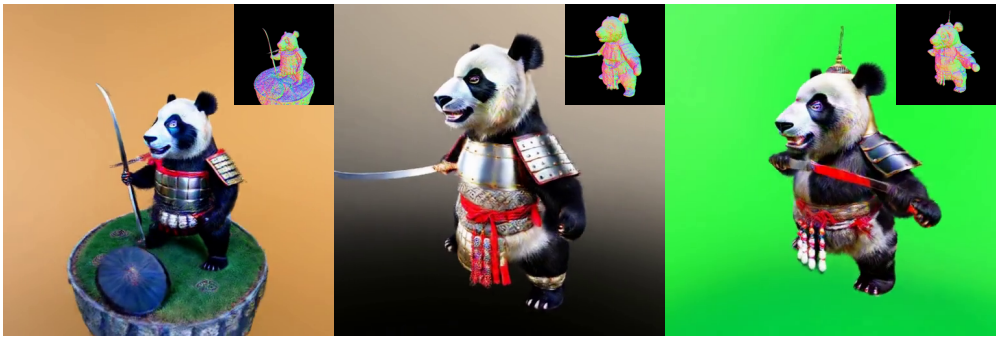
"A steampunk owl with mechanical wings"



"A cowboy raccoon with a lasso"



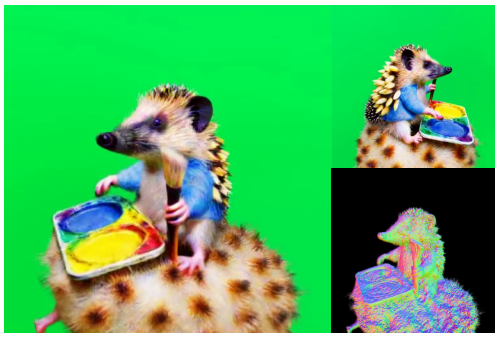
"A llama in a tuxedo at a fancy gala"



"A samurai panda in traditional armor"

Figure 10: **Diverse NeRF results of CFD distilling MVDream then Stable Diffusion [37] on Complex Prompts.**

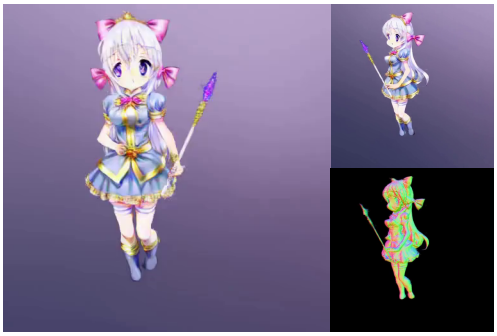
"A samurai panda with a bamboo sword"



"A painter hedgehog with a palette"

"A knight fox in shining armor"

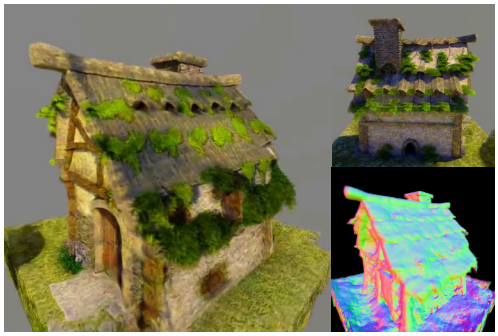
"A wizard frog with a spellbook"



"A manga magical girl with magic wand"

"A mecha vampire girl chibi"

"DSLR Camera, photography, dslr, camera, noobie, boxmodeling, maya"



"Medieval House, grass, medieval, vines, farm, middleage, medieval-house, stone, house, home, wood, medieval-decor"

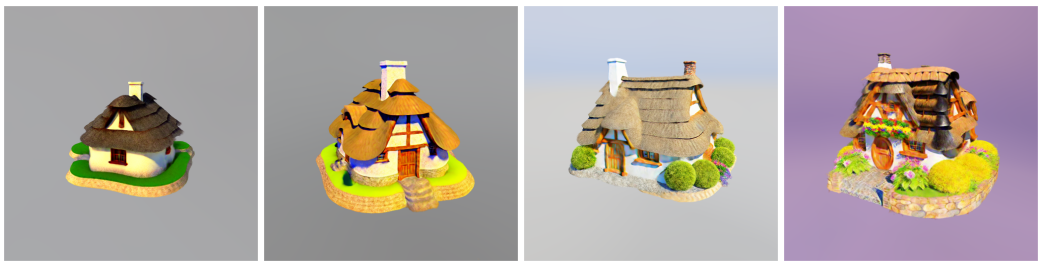
Figure 11: **NeRF results of CFD distilling MVDream then Stable Diffusion [37] on Complex Prompts.**

$\sqrt{1-\gamma}$	3D-FID (↓)	3D-IS (↑)
1.0	12.36	1.884±0.452
0.9999	**11.06**	1.889±0.388
0.999	11.58	**2.007±0.375**
0.99	13.87	1.967±0.484

Table 5: 3D generation quality and diversity with Stable Diffusion [37]. 8 seeds for each of the 5 different prompts.

C.2 Ablation on the Design Space

We ablate our proposed improvement step by step in this section. Timestep annealing [12, 44, 52] is helpful for forming finer details. Adding negative prompts [14, 27, 35] helps to improve generation styles. We also find that adding negative prompts is crucial when timestep $t(\tau)$ is small. Without negative prompts, the color of samples will become unnatural during the optimization at small timesteps. In this work, we apply negative prompts by directly replace the unconditional prediction of the diffusion model with prediction conditioned on negative prompts. Finally, by changing the random sampled noise in SDS with our multi-view consistent Gaussian noise, the generated samples can form much richer details and are more diverse. We visualize this ablation in Fig. 12.



(a) random timesteps (b) + annealing timesteps (c) + negative prompts (d) + consistent noise

Figure 12: **Ablation on the proposed improvements.**

D Experiment Details

3D-FID We compute the FID score between the rendered images for the generated 3D samples and the images generated by the teacher diffusion models following VSD [44]. Since MVDream [35] has relatively weak generation abilities, we choose to use the images generated by Stable Diffusion [37] as real image set when computing FID. Specifically, we use 2k images generated by Stable Diffusion for each of the prompts. For each of the generated 3D samples, we uniformly take 24 rendered views to compute the FID score. We mix the rendered images from different prompts, views and seeds as the fake image set. Finally, the real image set consists of 10k images and fake image set consists of 960 images. We use FID implementation from torchmetrics package with feature=64.

3D-IS We compute the IS for the front-view images on one set of samples corresponding to the same prompt. Then we average the IS across different prompts. We use IS implementation from torchmetrics package.

3D-CLIP We compute the CLIP cosine similarity between the rendered images of the 3D samples and the corresponding text prompt. For one sample, we render 120 views and take the maximum CLIP score. Then we average the CLIP score across different seeds and prompts. We use CLIP socre implementation from torchmetrics package.

Algorithm 1 CFD

Input: 3D representation parameter θ, prompt y, pre-trained diffusion model $\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, y)$, render $\boldsymbol{g}_\theta(\boldsymbol{c})$, annealing time-schedule $t(\tau)$, learning rate lr.
Output: 3D representation parameter θ.
for τ from 0 to τ_{end} **do**
  Sample camera view $\boldsymbol{c}$
  Render image $\boldsymbol{g}_\theta(\boldsymbol{c})$, depth map $\boldsymbol{Depth}(\boldsymbol{c})$, and opacity map $\boldsymbol{Opacity}(\boldsymbol{c})$
  Get diffusion timestep $t(\tau)$
  Compute 3D Consistent Noise $\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})$ ▷ Refer to Algorithm 2
  $\boldsymbol{x}_t \leftarrow \alpha_t \boldsymbol{g}_\theta(\boldsymbol{c}) + \sigma_t \tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})$
  $\theta \leftarrow \theta - lr \cdot (\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t(\tau), y) - \tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})) \frac{\partial \boldsymbol{g}_\theta(\boldsymbol{c})}{\partial \theta}$
end for

Algorithm 2 Computing 3D Consistent Noise

Initialization: Noise background $\boldsymbol{\epsilon}_{bg}$, high resolution noise $\boldsymbol{\epsilon}_{ref}$ in reference space E_{ref}, opacity threshold o_{th}, noise injection rate γ.
Input: Depth map $\boldsymbol{Depth}(\boldsymbol{c})$, opacity map $\boldsymbol{Opacity}(\boldsymbol{c})$.
Output: $\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c}) = \boldsymbol{\epsilon}_{out}$.
Triangulate the pixels to $\boldsymbol{p}$
Project those triangles to the surface $\boldsymbol{ctw}(\boldsymbol{p})$ in world space E_{world} according to $\boldsymbol{Depth}(\boldsymbol{c})$
Warp the triangles from world space E_{world} to reference space E_{ref} as $\mathcal{T}^{-1}(\boldsymbol{ctw}_{\boldsymbol{c}}(\boldsymbol{p}))$
Rasterize and aggregate the noise values on $\boldsymbol{\epsilon}_{ref}$ corvered by the trangles
$\boldsymbol{\epsilon}_{out} \leftarrow \boldsymbol{\epsilon}_{bg}$
$\boldsymbol{\epsilon}_{out}[\boldsymbol{Opacity}(\boldsymbol{c}) > o_{th}]_{\boldsymbol{p}} \leftarrow \frac{1}{\sqrt{n}} \sum^{n}_{(x,y)_i \text{ covered by the rasterized triangle } \mathcal{T}^{-1}(\boldsymbol{ctw}_{\boldsymbol{c}}(\boldsymbol{p}))} \boldsymbol{\epsilon}_{ref}[x, y]$
if $\gamma > 0$ **then**
  $\boldsymbol{\epsilon}_{bg} \leftarrow \sqrt{1-\gamma}\boldsymbol{\epsilon}_{bg} + \sqrt{\gamma} \cdot \text{randn\_like}(\boldsymbol{\epsilon}_{bg})$ ▷ SDE noise injection
  $\boldsymbol{\epsilon}_{ref} \leftarrow \sqrt{1-\gamma}\boldsymbol{\epsilon}_{ref} + \sqrt{\gamma} \cdot \text{randn\_like}(\boldsymbol{\epsilon}_{ref})$
end if
Return $\boldsymbol{\epsilon}_{out}$

E Algorithms

We provide pseudo algorithms for CFD in Algorithm 1. Algorithm 2 presents how to compute the multi-view consistent Gaussian noise $\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})$.

Choices of warping function $\mathcal{T}^{-1}$ and reference space E_{ref} We choose E_{ref} to be a 2D square space $E_{ref} = [-1, 1]^2$ in this work to utilize existing fast rasterization algorithms, so that Algorithm 2 can be efficiently computed. We design a warping function $\mathcal{T}^{-1}$ to map points in 3D world space E_{world} to 2D reference space E_{ref}. Specifically, to compute the warping $\mathcal{T}^{-1}$ we first convert the points at (x_p, y_p, z_p) to spherical coordinates (r_p, θ_p, ϕ_p). For simplicity, we only present the case when $\phi_p \in [0, \frac{\pi}{2})$. The point is then mapped to $(x_r, y_r) \in E_{ref}$, where

$$\begin{cases} x_r = \sqrt{1 - \cos\theta_p}, \\ y_r = \sqrt{1 - \cos\theta_p} \cdot (2 \cdot \frac{\phi_p}{\frac{\pi}{2}} - 1). \end{cases} \tag{15}$$

Under this mapping function, one can verify that $\mathrm{d}x_r \mathrm{d}y_r = |\frac{\partial(x_r, y_r)}{\partial(\theta_p, \phi_p)}| \mathrm{d}\theta_p \mathrm{d}\phi_p = \frac{2}{\pi} \cdot \sin\theta_p \mathrm{d}\theta_p \mathrm{d}\phi_p$. So points uniformly scattered on the sphere in 3D space E_{world} will remain uniform after being mapped to the reference 2D space E_{ref}. This design helps to improve the fairness of Algorithm 2 so that we can use a lower resolution reference space while keeping most of the warped triangles covering enough area in the reference space E_{ref}. Notably, two different triangles could overlap with the warping defined by Eq. 15, resulting in correlations across the pixels of the computed noise function $\tilde{\boldsymbol{\epsilon}}(\theta, \boldsymbol{c})$ in the same camera view. This overlap occurs only when the surface of the 3D object intersects the radius of a sphere centered at the origin of the E_{world} more than once. However, we do not observe the destructive effects seen in other interpolation methods that can lead to correlation

between pixels (as in Fig. 6 (b)) in our experiments, and we believe it is unnecessary to find a warping function that avoids such overlapping completely.

Reference space E_{ref} resolution We use reference space with resolution of 2048×2048 in most of our experiments. This will only introduce approximately 5% computation overhead to our training (tested on RTX-3090). The teacher model Stable Diffusion represents the whole object with latent at 64 resolution and MVDream [35] 32, so noise map with 2048 resolution is sufficient. We also observe the quality is similar with noise map resolutions from 512 to 2048.

F Clean Flow SDE

F.1 Background

Song et al. [38] presented a SDE that has the same marginal distribution $p_t(\boldsymbol{x}_t)$ as the forward diffusion process (Eq. 1). EDM [13] presented a more general form of this SDE, and the SDE corresponds to forward process defined in Eq. 1 takes the following form:

$$\mathrm{d}\big(\frac{\boldsymbol{x}_\pm}{\alpha_t}\big) = -\sigma_t \nabla_{\boldsymbol{x}_\pm} \log p_t(\boldsymbol{x}_\pm)\mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big) \pm \beta_t\big(\frac{\sigma_t}{\alpha_t}\big)\sigma_t \nabla_{\boldsymbol{x}_\pm} \log p_t(\boldsymbol{x}_\pm)\mathrm{d}t + \sqrt{2\beta_t}\big(\frac{\sigma_t}{\alpha_t}\big)\mathrm{d}\boldsymbol{w}_t \tag{16}$$

$$= \big(\mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big) \mp \frac{\sigma_t}{\alpha_t}\beta_t \mathrm{d}t\big) \cdot \underbrace{\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_\pm, t, y)}_{-\sigma_t \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})} + \sqrt{2\beta_t}\big(\frac{\sigma_t}{\alpha_t}\big)\mathrm{d}\boldsymbol{w}_t, \tag{17}$$

where $\mathrm{d}\boldsymbol{w}_t$ is the standard Wiener process. If we set $\alpha_t = 1$ for all $t \in [0, T]$, Eq. 17 will become the same SDE in EDM [13]. The initial condition for the forward process is $\boldsymbol{x}_+ \sim p_{t_s}(\boldsymbol{x}_+)$ at $t = t_s$ (t_s is small enough but $t_s > 0$ to avoid numerical issues), and for the reverse process, it is $\boldsymbol{x}_- \sim \mathcal{N}(\mathbf{0}, \sigma_T^2 \boldsymbol{I})$ at $t = T$ (Note that we also let α_T be a small number but $\alpha_T > 0$ to avoid numerical issues).

F.2 Clean Flow SDE

The clean flow SDE takes the following form:

$$\begin{cases} \mathrm{d}\hat{\boldsymbol{x}}^{\mathrm{c}}_\pm = \big(\mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big) \mp \frac{\sigma_t}{\alpha_t}\beta_t \mathrm{d}t\big) \cdot \big(\boldsymbol{\epsilon}_\phi(\alpha_t \hat{\boldsymbol{x}}^{\mathrm{c}}_\pm + \sigma_t \tilde{\boldsymbol{\epsilon}}_\pm, t, y) - \tilde{\boldsymbol{\epsilon}}_\pm\big), \\ \mathrm{d}\tilde{\boldsymbol{\epsilon}}_\pm = \mp \tilde{\boldsymbol{\epsilon}}_\pm \beta_t \mathrm{d}t + \sqrt{2\beta_t}\mathrm{d}\boldsymbol{w}_t, \end{cases} \tag{18}$$

where $\mathrm{d}\boldsymbol{w}_t$ is the standard Wiener process. For the forward process, the initial condition at $t = t_s$ is $\hat{\boldsymbol{x}}^{\mathrm{c}}_+ \sim p_0(\boldsymbol{x}_+)$, $\tilde{\boldsymbol{\epsilon}}_+ \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, and $\hat{\boldsymbol{x}}^{\mathrm{c}}_+$ and $\tilde{\boldsymbol{\epsilon}}_+$ are independent. For the reverse process, the initial condition at $t = T$ is $\hat{\boldsymbol{x}}^{\mathrm{c}}_- = \mathbf{0}$ and $\tilde{\boldsymbol{\epsilon}}_- \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

Proposition 1 (Clean flow SDE is equivalent to diffusion SDE)**.** *In Eq. 18, if we define a new variable $\boldsymbol{x}'_\pm$ according to*

$$\boldsymbol{x}'_\pm = \alpha_t \hat{\boldsymbol{x}}^{c}_\pm + \sigma_t \tilde{\boldsymbol{\epsilon}}_\pm, \tag{19}$$

then $\boldsymbol{x}'_\pm$ and $\boldsymbol{x}_\pm$ in Eq. 17 have the same law (probability distribution) for all $t \in [t_s, T]$. i.e. Eq. 18 and Eq. 17 are equivalent.

proof. We prove the equivalence by showing that the initial conditions and dynamics for $\boldsymbol{x}'_\pm$ and $\boldsymbol{x}_\pm$ are identical.

Initial conditions. For the forward process of Eq. 18 at $t = t_s$, $\boldsymbol{x}'_+ = \alpha_{t_s} \hat{\boldsymbol{x}}^{\mathrm{c}}_+ + \sigma_{t_s} \tilde{\boldsymbol{\epsilon}}_+$. Thus, $\boldsymbol{x}'_\pm \sim p_{t_s}(\boldsymbol{x}'_\pm)$ according to the definition of a forward diffusion process (Eq. 1). For the reverse process of Eq. 18 at $t = T$, $\boldsymbol{x}'_- = \alpha_T \cdot \mathbf{0} + \sigma_T \tilde{\boldsymbol{\epsilon}}_- = \sigma_T \tilde{\boldsymbol{\epsilon}}_-$. So $\boldsymbol{x}'_- \sim \mathcal{N}(\mathbf{0}, \sigma_T^2 \boldsymbol{I})$.

Dynamics. The dynamic of $\boldsymbol{x}'_{\pm}$ can be derived according to:

$$
\begin{aligned}
\mathrm{d}\big(\frac{\boldsymbol{x}'_{\pm}}{\alpha_t}\big) =& \mathrm{d}\big(\hat{\boldsymbol{x}}^{\mathrm{c}}_{\pm} + \frac{\sigma_t}{\alpha_t}\tilde{\boldsymbol{\epsilon}}_{\pm}\big) \\
=& \mathrm{d}\hat{\boldsymbol{x}}^{\mathrm{c}}_{\pm} + \mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big)\tilde{\boldsymbol{\epsilon}}_{\pm} + \frac{\sigma_t}{\alpha_t}\mathrm{d}\tilde{\boldsymbol{\epsilon}}_{\pm} \\
=& \big(\mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big) \mp \frac{\sigma_t}{\alpha_t}\beta_t\mathrm{d}t\big) \cdot \big(\boldsymbol{\epsilon}_\phi(\alpha_t\hat{\boldsymbol{x}}^{\mathrm{c}}_{\pm} + \sigma_t\tilde{\boldsymbol{\epsilon}}_{\pm}, t, y) - \tilde{\boldsymbol{\epsilon}}_{\pm}\big) + \mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big)\tilde{\boldsymbol{\epsilon}}_{\pm} + \frac{\sigma_t}{\alpha_t}\mathrm{d}\tilde{\boldsymbol{\epsilon}}_{\pm} \\
=& \big(\mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big) \mp \frac{\sigma_t}{\alpha_t}\beta_t\mathrm{d}t\big) \cdot \big(\boldsymbol{\epsilon}_\phi(\boldsymbol{x}'_{\pm}, t, y) - \tilde{\boldsymbol{\epsilon}}_{\pm}\big) + \mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big)\tilde{\boldsymbol{\epsilon}}_{\pm} + \frac{\sigma_t}{\alpha_t}\mathrm{d}\tilde{\boldsymbol{\epsilon}}_{\pm} \\
=& \big(\mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big) \mp \frac{\sigma_t}{\alpha_t}\beta_t\mathrm{d}t\big) \cdot \boldsymbol{\epsilon}_\phi(\cdots) - \mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big)\tilde{\boldsymbol{\epsilon}}_{\pm} \pm \frac{\sigma_t}{\alpha_t}\beta_t\tilde{\boldsymbol{\epsilon}}_{\pm}\mathrm{d}t + \mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big)\tilde{\boldsymbol{\epsilon}}_{\pm} + \frac{\sigma_t}{\alpha_t}\mathrm{d}\tilde{\boldsymbol{\epsilon}}_{\pm} \\
=& \big(\mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big) \mp \frac{\sigma_t}{\alpha_t}\beta_t\mathrm{d}t\big) \cdot \boldsymbol{\epsilon}_\phi(\cdots) \pm \frac{\sigma_t}{\alpha_t}\beta_t\tilde{\boldsymbol{\epsilon}}_{\pm}\mathrm{d}t + \frac{\sigma_t}{\alpha_t}\mathrm{d}\tilde{\boldsymbol{\epsilon}}_{\pm} \\
=& \big(\mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big) \mp \frac{\sigma_t}{\alpha_t}\beta_t\mathrm{d}t\big) \cdot \boldsymbol{\epsilon}_\phi(\cdots) \pm \frac{\sigma_t}{\alpha_t}\beta_t\tilde{\boldsymbol{\epsilon}}_{\pm}\mathrm{d}t \mp \frac{\sigma_t}{\alpha_t}\tilde{\boldsymbol{\epsilon}}_{\pm}\beta_t\mathrm{d}t + \sqrt{2\beta_t}\big(\frac{\sigma_t}{\alpha_t}\big)\mathrm{d}\boldsymbol{w}_t \\
=& \big(\mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big) \mp \frac{\sigma_t}{\alpha_t}\beta_t\mathrm{d}t\big) \cdot \boldsymbol{\epsilon}_\phi(\boldsymbol{x}'_{\pm}, t, y) + \sqrt{2\beta_t}\big(\frac{\sigma_t}{\alpha_t}\big)\mathrm{d}\boldsymbol{w}_t.
\end{aligned}
\tag{20}
$$

So $\boldsymbol{x}'_{\pm}$ and $\boldsymbol{x}_{\pm}$ follow the same dynamics. □

We present a stochastic sampler in Algo. 3 that is equivalent Algorithm 2 in EDM [13] to show a practice implementation of Eq. 20 for sampling.

Algorithm 3 A SDE sampler that is equivalent to Algorithm 2 in EDM [13]

Input: Diffusion model (sample prediction) D_ϕ, $t_{i\in\{0,\cdots,N\}}$, $\gamma_{i\in\{0,\cdots,N-1\}}$, S_{noise}.
Output: $\hat{\boldsymbol{x}}^{\mathrm{c}}_N$.
Initialize $\tilde{\boldsymbol{\epsilon}}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, $\hat{\boldsymbol{x}}^{\mathrm{c}}_0 = \mathbf{0}$
for $i \in \{0, \cdots, N-1\}$ **do**
  Sample $\boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, S^2_{\text{noise}}\boldsymbol{I})$
  $\hat{t}_i \leftarrow t_i + \gamma_i t_i$
  $\tilde{\boldsymbol{\epsilon}}_{i+1} \leftarrow \frac{t_i}{\hat{t}_i}\tilde{\boldsymbol{\epsilon}}_i + \sqrt{1 - (\frac{t_i}{\hat{t}_i})^2}\boldsymbol{\epsilon}_i$
  $\boldsymbol{d}_i \leftarrow (\hat{\boldsymbol{x}}^{\mathrm{c}}_i - D_\phi(\hat{\boldsymbol{x}}^{\mathrm{c}}_i + \hat{t}_i\tilde{\boldsymbol{\epsilon}}_{i+1}, \hat{t}_i))/\hat{t}_i$
  $\hat{\boldsymbol{x}}^{\mathrm{c}}_{i+1} \leftarrow \hat{\boldsymbol{x}}^{\mathrm{c}}_i + (t_{i+1} - \hat{t}_i)\boldsymbol{d}_i$
  if $t_{i+1} \neq 0$ **then**
    $\boldsymbol{d}'_i \leftarrow (\hat{\boldsymbol{x}}^{\mathrm{c}}_{i+1} - D_\phi(\hat{\boldsymbol{x}}^{\mathrm{c}}_{i+1} + t_{i+1}\tilde{\boldsymbol{\epsilon}}_{i+1}, t_{i+1}))/t_{i+1}$
    $\hat{\boldsymbol{x}}^{\mathrm{c}}_{i+1} \leftarrow \hat{\boldsymbol{x}}^{\mathrm{c}}_i + (t_{i+1} - \hat{t}_i)(\frac{1}{2}\boldsymbol{d}_i + \frac{1}{2}\boldsymbol{d}'_i)$ ▷ Apply 2nd order correction
  end if
end for
Return $\hat{\boldsymbol{x}}^{\mathrm{c}}_N$

F.3 Properties of $\hat{\boldsymbol{x}}^{\mathbf{c}}_{\pm}$

F.3.1 $\hat{\boldsymbol{x}}^{\mathbf{c}}_t$ are Clean Images for all $t \in [t_s, T]$

Lemma 1 (Sample predictions are non-noisy images)**.** *The sample prediction of the diffusion model*

$$
\hat{\boldsymbol{x}}^{gt}_t \triangleq \frac{\boldsymbol{x}_t - \sigma_t\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, y)}{\alpha_t} \tag{21}
$$

is a weighted average of images in the target distribution $p_0(\boldsymbol{x}_0)$:

$$
\hat{\boldsymbol{x}}^{gt}_t = \mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t]. \tag{22}
$$

Thus, $\hat{\boldsymbol{x}}^{gt}_t$ are non-noisy images. Furthermore,

$$
\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, y) = \frac{\boldsymbol{x}_t - \alpha_t\mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t]}{\sigma_t}. \tag{23}
$$

proof.

$$
\begin{aligned}
\hat{\boldsymbol{x}}_t^{\text{gt}} =& \frac{\boldsymbol{x}_t - \sigma_t \boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, y)}{\alpha_t} \\
=& \frac{1}{\alpha_t}\big(\boldsymbol{x}_t + \sigma_t^2 \nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t)\big) \\
=& \frac{1}{\alpha_t}\big(\boldsymbol{x}_t + \frac{\sigma_t^2}{p_t(\boldsymbol{x}_t)} \nabla_{\boldsymbol{x}_t} p_t(\boldsymbol{x}_t)\big) \\
=& \frac{1}{\alpha_t}\big(\boldsymbol{x}_t + \frac{\sigma_t^2}{p_t(\boldsymbol{x}_t)} \nabla_{\boldsymbol{x}_t} \int p(\boldsymbol{x}_t|\boldsymbol{x}_0) p_0(\boldsymbol{x}_0) \mathrm{d}\boldsymbol{x}_0\big) \\
=& \frac{1}{\alpha_t}\big(\boldsymbol{x}_t + \frac{\sigma_t^2}{p_t(\boldsymbol{x}_t)} \int p(\boldsymbol{x}_t|\boldsymbol{x}_0) \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|\boldsymbol{x}_0) p_0(\boldsymbol{x}_0) \mathrm{d}\boldsymbol{x}_0\big) \\
=& \frac{1}{\alpha_t}\big(\boldsymbol{x}_t + \frac{\sigma_t^2}{p_t(\boldsymbol{x}_t)} \int p(\boldsymbol{x}_t|\boldsymbol{x}_0) \nabla_{\boldsymbol{x}_t} \big(-\frac{(\boldsymbol{x}_t - \alpha_t \boldsymbol{x}_0)^2}{2\sigma_t^2}\big) p_0(\boldsymbol{x}_0) \mathrm{d}\boldsymbol{x}_0\big) \\
=& \frac{1}{\alpha_t}\big(\boldsymbol{x}_t - \frac{1}{p_t(\boldsymbol{x}_t)} \int p(\boldsymbol{x}_t|\boldsymbol{x}_0)(\boldsymbol{x}_t - \alpha_t \boldsymbol{x}_0) p_0(\boldsymbol{x}_0) \mathrm{d}\boldsymbol{x}_0\big) \\
=& \frac{1}{\alpha_t}\big(\boldsymbol{x}_t - \boldsymbol{x}_t \frac{\int p(\boldsymbol{x}_t|\boldsymbol{x}_0) p_0(\boldsymbol{x}_0) \mathrm{d}\boldsymbol{x}_0}{p_t(\boldsymbol{x}_t)} + \alpha_t \int \boldsymbol{x}_0 \frac{p(\boldsymbol{x}_t|\boldsymbol{x}_0) p_0(\boldsymbol{x}_0)}{p_t(\boldsymbol{x}_t)} \mathrm{d}\boldsymbol{x}_0\big) \\
=& \frac{1}{\alpha_t}\big(\boldsymbol{x}_t - \boldsymbol{x}_t + \alpha_t \int \boldsymbol{x}_0 p(\boldsymbol{x}_0|\boldsymbol{x}_t) \mathrm{d}\boldsymbol{x}_0\big) \\
=& \int \boldsymbol{x}_0 p(\boldsymbol{x}_0|\boldsymbol{x}_t) \mathrm{d}\boldsymbol{x}_0 \\
=& \mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t].
\end{aligned} \tag{24}
$$

Thus,

$$
\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, y) = \frac{\boldsymbol{x}_t - \alpha_t \hat{\boldsymbol{x}}_t^{\text{gt}}}{\sigma_t} = \frac{\boldsymbol{x}_t - \alpha_t \mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t]}{\sigma_t}. \tag{25}
$$

□

Proposition 2 ($\hat{\boldsymbol{x}}_\pm^{\text{c}}$ are non-noisy images)**.** *$\hat{\boldsymbol{x}}_\pm^{c}$ in Eq. 18 are non-noisy images for all $t \in [t_s, T]$.*

proof. Since the initial conditions of $\hat{\boldsymbol{x}}_\pm^{\text{c}}$ ($\hat{\boldsymbol{x}}_-^{\text{c}} = \mathbf{0}$ for reverse process and $\hat{\boldsymbol{x}}_+^{\text{c}} \sim p_0(\boldsymbol{x}_0)$ for forward process) implies $\hat{\boldsymbol{x}}_\pm^{\text{c}}$ are initialized as non-noisy images, we only need to show that the dynamic of $\hat{\boldsymbol{x}}_\pm^{\text{c}}$ will not introduce Gaussian noise into $\hat{\boldsymbol{x}}_\pm^{\text{c}}$.

The dynamic of $\hat{\boldsymbol{x}}_\pm^{\text{c}}$ can be reformulated as:

$$
\begin{aligned}
\mathrm{d}\hat{\boldsymbol{x}}_\pm^{\text{c}} =& \big(\mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big) \mp \frac{\sigma_t}{\alpha_t}\beta_t \mathrm{d}t\big) \cdot \big(\boldsymbol{\epsilon}_\phi(\alpha_t \hat{\boldsymbol{x}}_\pm^{\text{c}} + \sigma_t \tilde{\boldsymbol{\epsilon}}_\pm, t, y) - \tilde{\boldsymbol{\epsilon}}_\pm\big), \\
=& \big(\mathrm{d}\big(\frac{\sigma_t}{\alpha_t}\big) \mp \frac{\sigma_t}{\alpha_t}\beta_t \mathrm{d}t\big) \cdot \frac{\alpha_t \hat{\boldsymbol{x}}_\pm^{\text{c}} + \sigma_t \tilde{\boldsymbol{\epsilon}}_\pm - \alpha_t \mathbb{E}[\boldsymbol{x}_0|\alpha_t \hat{\boldsymbol{x}}_\pm^{\text{c}} + \sigma_t \tilde{\boldsymbol{\epsilon}}_\pm] - \sigma_t \tilde{\boldsymbol{\epsilon}}_\pm}{\sigma_t}, \\
=& \big(\mathrm{d}\big(\log \frac{\sigma_t}{\alpha_t}\big) \mp \beta_t \mathrm{d}t\big) \cdot (\hat{\boldsymbol{x}}_\pm^{\text{c}} - \mathbb{E}[\boldsymbol{x}_0|\alpha_t \hat{\boldsymbol{x}}_\pm^{\text{c}} + \sigma_t \tilde{\boldsymbol{\epsilon}}_\pm]).
\end{aligned} \tag{26}
$$

As Eq. 26 shows that $\hat{\boldsymbol{x}}_\pm^{\text{c}}$ is always moving towards non-noisy sample prediction $\hat{\boldsymbol{x}}_t^{\text{gt}} = \mathbb{E}[\boldsymbol{x}_0|\boldsymbol{x}_t]$ for all $t \in [t_s, T]$, $\hat{\boldsymbol{x}}_\pm^{\text{c}}$ will be non-noisy for all $t \in [t_s, T]$. □

We also visualize $\hat{\boldsymbol{x}}_\pm^{\text{c}}$ at random timestep $t \in [0, T]$ of Stable Diffusion [37] sampling processes in Appx. Fig. 15 to show that they are visually clean (non-noisy). We use clean variable to refer to $\hat{\boldsymbol{x}}_\pm^{\text{c}}$ in this work since it is always non-noisy.

F.3.2 Initialization of $\hat{\boldsymbol{x}}_t^{\text{c}}$

The initial condition of reverse-time clean flow SDE (Eq. 18) is given by $\boldsymbol{x}_- = \mathbf{0}$ and $\tilde{\boldsymbol{\epsilon}}_- \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. This is consistent with a typical initialization of NeRF: the whole scene of the NeRF being all grey.

When we set $\hat{\boldsymbol{x}}^{\mathrm{c}}_{-} = \mathbf{0}$ as the initial condition for the clean flow SDE, it corresponds to the initial condition $\boldsymbol{x}_{-} \sim \mathcal{N}(\mathbf{0}, \sigma_T^2 \boldsymbol{I})$ [13] in the diffusion SDE (Eq. 17). However, since we set a small nonzero α_T at the beginning, the strict initial condition of the diffusion SDE should be $p_T(\boldsymbol{x}_T)$, which is slightly different from $\mathcal{N}(\mathbf{0}, \sigma_T^2 \boldsymbol{I})$. In this case, we should set $\hat{\boldsymbol{x}}^{\mathrm{c}}_{-} \sim p_0(\boldsymbol{x}_0)$ in the clean flow SDE to make the initial condition of the two SDE identical. Prior works usually ignore the small difference between $p_T(\boldsymbol{x}_T)$ and $\mathcal{N}(\mathbf{0}, \sigma_T^2 \boldsymbol{I})$ and starts from pure noise when sampling [20], and from our practical observation, given different initial $\hat{\boldsymbol{x}}^{\mathrm{c}}_{-} \neq \mathbf{0}$ but the same $\tilde{\boldsymbol{\epsilon}}_{-}$, clean flow SDE will yield almost identical outputs (given the same seeds), which implies the endpoints of $\hat{\boldsymbol{x}}^{\mathrm{c}}_t$ are not sensitive to initialization of $\hat{\boldsymbol{x}}^{\mathrm{c}}_t$. So we choose to set $\hat{\boldsymbol{x}}^{\mathrm{c}}_{-} = \mathbf{0}$ in this work as the initial condition.

F.3.3 Endpoints of $\hat{\boldsymbol{x}}^{\mathrm{c}}_t$

At the end of the reverse-time clean flow SDE, $\hat{\boldsymbol{x}}^{\mathrm{c}}_{-} = \boldsymbol{x}_0 \sim p_0(\boldsymbol{x}_0)$. So $\hat{\boldsymbol{x}}^{\mathrm{c}}_t$ also ends as a sample in the target distribution $p_0(\boldsymbol{x}_0)$ as $\boldsymbol{x}_0$ in the reverse-time diffusion SDE.

F.4 Properties of $\tilde{\boldsymbol{\epsilon}}_{\pm}$

$\tilde{\boldsymbol{\epsilon}}_{\pm}$ can be seen as the "pure noise" part in the clean flow SDE (Eq. 18). Notably, the evolution of $\tilde{\boldsymbol{\epsilon}}_{\pm}$ does not depend on $\hat{\boldsymbol{x}}^{\mathrm{c}}_t$ and has a closed-form solution. The dynamic of $\tilde{\boldsymbol{\epsilon}}_{\pm}$ is given by

$$\mathrm{d}\tilde{\boldsymbol{\epsilon}}_{\pm} = \mp \tilde{\boldsymbol{\epsilon}}_{\pm} \beta_t \mathrm{d}t + \sqrt{2\beta_t} \mathrm{d}\boldsymbol{w}_t. \tag{27}$$

The initial condition for $\tilde{\boldsymbol{\epsilon}}_{\pm}$ in both the forward and reverse process are $\tilde{\boldsymbol{\epsilon}}_{\pm} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

F.4.1 Closed-form Solutions

For the forward process,

$$\begin{aligned}
\mathrm{d}\big(e^{\int_0^t \beta_s \mathrm{d}s} \tilde{\boldsymbol{\epsilon}}_{+}\big) &= e^{\int_0^t \beta_s \mathrm{d}s} \mathrm{d}\tilde{\boldsymbol{\epsilon}}_{+} + \tilde{\boldsymbol{\epsilon}}_{+} \beta_t e^{\int_0^t \beta_s \mathrm{d}s} \mathrm{d}t \\
&= e^{\int_0^t \beta_s \mathrm{d}s} \sqrt{2\beta_t} \mathrm{d}\boldsymbol{w}_t - \tilde{\boldsymbol{\epsilon}}_{+} \beta_t e^{\int_0^t \beta_s \mathrm{d}s} \mathrm{d}t \; + \tilde{\boldsymbol{\epsilon}}_{+} \beta_t e^{\int_0^t \beta_s \mathrm{d}s} \mathrm{d}t \\
&= e^{\int_0^t \beta_s \mathrm{d}s} \sqrt{2\beta_t} \mathrm{d}\boldsymbol{w}_t.
\end{aligned} \tag{28}$$

Integral on both side of Eq. 28, we have

$$e^{\int_0^t \beta_s \mathrm{d}s} \tilde{\boldsymbol{\epsilon}}_{+} - \tilde{\boldsymbol{\epsilon}}_0 = \int_0^t \sqrt{2\beta_s} e^{\int_0^s \beta_r \mathrm{d}r} \mathrm{d}\boldsymbol{w}_s. \tag{29}$$

Thus, we obtain the solution of $\tilde{\boldsymbol{\epsilon}}_{+}$:

$$\tilde{\boldsymbol{\epsilon}}_{+} = e^{-\int_0^t \beta_s \mathrm{d}s} \cdot \tilde{\boldsymbol{\epsilon}}_0 + e^{-\int_0^t \beta_s \mathrm{d}s} \int_0^t \sqrt{2\beta_s} e^{\int_0^s \beta_r \mathrm{d}r} \mathrm{d}\boldsymbol{w}_s. \tag{30}$$

Similarly, we can obtain the solution of $\tilde{\boldsymbol{\epsilon}}_{-}$:

$$\tilde{\boldsymbol{\epsilon}}_{-} = e^{-\int_t^T \beta_s \mathrm{d}s} \cdot \tilde{\boldsymbol{\epsilon}}_T + e^{-\int_t^T \beta_s \mathrm{d}s} \int_T^t \sqrt{2\beta_s} e^{\int_s^T \beta_r \mathrm{d}r} \mathrm{d}\bar{\boldsymbol{w}}_s. \tag{31}$$

Specifically, we can derive a closed-form formulation to compute $\tilde{\boldsymbol{\epsilon}}_{+}(t)$ given $\tilde{\boldsymbol{\epsilon}}_{+}(t')$ for $t' < t$ from Eq. 30, which takes the following form:

$$\tilde{\boldsymbol{\epsilon}}_{+}(t) = e^{-\int_{t'}^t \beta_s \mathrm{d}s} \tilde{\boldsymbol{\epsilon}}_{+}(t') + \sqrt{1 - e^{-2\int_{t'}^t \beta_s \mathrm{d}s}} \boldsymbol{\epsilon}, \tag{32}$$

$$= \sqrt{1-\gamma} \tilde{\boldsymbol{\epsilon}}_{+}(t') + \sqrt{\gamma} \boldsymbol{\epsilon}, \tag{33}$$

where $\gamma = 1 - e^{-2\int_{t'}^t \beta_s \mathrm{d}s}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

For $\tilde{\boldsymbol{\epsilon}}_{-}(t)$ and $t' > t$,

$$\tilde{\boldsymbol{\epsilon}}_{-}(t) = e^{-\int_t^{t'} \beta_s \mathrm{d}s} \tilde{\boldsymbol{\epsilon}}_{-}(t') + \sqrt{1 - e^{-2\int_t^{t'} \beta_s \mathrm{d}s}} \boldsymbol{\epsilon}, \tag{34}$$

$$= \sqrt{1-\gamma} \tilde{\boldsymbol{\epsilon}}_{-}(t') + \sqrt{\gamma} \boldsymbol{\epsilon}, \tag{35}$$

where $\gamma = 1 - e^{-2\int_t^{t'} \beta_s \mathrm{d}s}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

F.4.2 $\tilde{\epsilon}_{\pm}$ is of Unit Variance for all $t \in [0, T]$

$$
\begin{aligned}
Var(\tilde{\boldsymbol{\epsilon}}_+) &= e^{-2\int_0^t \beta_s \mathrm{d}s} + e^{-2\int_0^t \beta_s \mathrm{d}s} \int_0^t 2\beta_s e^{2\int_0^s \beta_r \mathrm{d}r} \mathrm{d}s \\
&= e^{-2\int_0^t \beta_s \mathrm{d}s} (1 + \int_0^t 2\beta_s e^{2\int_0^s \beta_r \mathrm{d}r} \mathrm{d}s) \\
&= e^{-2\int_0^t \beta_s \mathrm{d}s} (1 + \int_0^t \mathrm{d}e^{2\int_0^s \beta_r \mathrm{d}r}) \\
&= e^{-2\int_0^t \beta_s \mathrm{d}s} (1 + e^{2\int_0^t \beta_r \mathrm{d}r} - 1) \\
&= 1,
\end{aligned} \tag{36}
$$

$$
\begin{aligned}
Var(\tilde{\boldsymbol{\epsilon}}_-) &= e^{-2\int_t^T \beta_s \mathrm{d}s} + e^{-2\int_t^T \beta_s \mathrm{d}s} \int_t^T 2\beta_s e^{2\int_s^T \beta_r \mathrm{d}r} \mathrm{d}s \\
&= e^{-2\int_t^T \beta_s \mathrm{d}s} (1 + \int_t^T 2\beta_s e^{2\int_s^T \beta_r \mathrm{d}r} \mathrm{d}s) \\
&= e^{-2\int_t^T \beta_s \mathrm{d}s} (1 - \int_t^T \mathrm{d}e^{2\int_s^T \beta_r \mathrm{d}r}) \\
&= e^{-2\int_t^T \beta_s \mathrm{d}s} (1 - 1 + e^{2\int_t^T \beta_r \mathrm{d}r}) \\
&= 1.
\end{aligned} \tag{37}
$$

F.5 Clean Flow ODE

When we set $\beta_t = 0$ in clean flow SDE (Eq. 18), it becomes determined and changes to an ODE (Eq. 8). Furthermore, $\mathrm{d}\tilde{\boldsymbol{\epsilon}}_{\pm} = 0$ and thus $\tilde{\boldsymbol{\epsilon}}_{\pm}$ will become a constant $\tilde{\boldsymbol{\epsilon}}$. This ODE is the same ODE presented in FSD [48]. It is also equivalent to the signal-ODE presented in BOOT [7] when the diffusion model is changed to sample-prediction.

G Discussion on the Choice of the Variable Space

G.1 Ground-truth Variable

Figure 13: **Compare with ISM [18].** Our methods can achieve competitive results with ISM while being more efficient.

Apart from the clean variable $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$, FSD [48] also defined another variable space that is visually clean, which is the *ground-truth variable* $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$. $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$ is defined by

$$
\hat{\boldsymbol{x}}_t^{\mathrm{gt}} \triangleq \frac{\boldsymbol{x}_t - \sigma_t \boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, y)}{\alpha_t}. \tag{38}
$$

$\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$ is also known as the "sample prediction" of the diffusion model. The ODE on $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$ is given by:

$$
\mathrm{d}\hat{\boldsymbol{x}}_t^{\mathrm{gt}} = -(\frac{\sigma_t}{\alpha_t}) \cdot \mathrm{d}\boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, y). \tag{39}
$$

Figure 14: **Visualization of noisy variable x_t.**

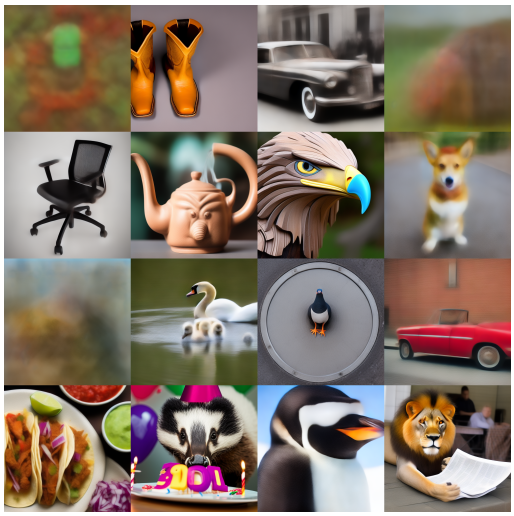


Figure 15: **Visualization of clean variable $\hat{x}_t^{\mathrm{c}}$.**

Concurrent work SDI [25] shares an insight similar to ours by also using rendered images to replace the "non-noisy variables" to guide 3D generation. The difference between SDI [25] and our method is that SDI replaced the ground-truth variable $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$ with rendered image $\boldsymbol{g}_\theta(\boldsymbol{c})$ but we replace the clean variable $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ with $\boldsymbol{g}_\theta(\boldsymbol{c})$.

Theoretically speaking, if it's just to solve the OOD problem when using image PF-ODE as a guidance for 3D generation, we think it's both reasonable to replace $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$ and $\hat{\boldsymbol{x}}_t^{\mathrm{c}}$ with rendered images, since they are both non-noisy throughout the diffusion process (Lemma 1 and Proposition 2). However, it's difficult to exactly compute the update rule in Eq. 39 since $\boldsymbol{x}_t$ is required on right hand side of Eq. 39. In order to recover $\boldsymbol{x}_t$ given $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$, SDI needs to solve a fixed point equation, which is hard to be solved [25]. In practice, SDI use DDIM inversion and inteprate this as the approximated solution of the fixed point equation. Difficulties also appear in works that attempt to apply guidance on the ground-truth variable $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$ for conditional image generation, as seen in UGD [2] and FreeDoM [50]. In contrast, we can compute the evolution of $\hat{\boldsymbol{x}}_\pm^{\mathrm{c}}$ exactly according to Eq. 18 without the need to solve a fixed point equation.

Additionally, another recent work ISM [18] can also be viewed as replacing the ground-truth variable $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$ as discussed in SDI [25], since the main difference between ISM and SDI loss is whether to apply text condition when computing DDIM inversion. We compare our methods with ISM in their code base. Our method can achieve competitive results with ISM while being more efficient (ISM costs 78% more time than ours, tested on RTX-3090 with batch size 1).

G.2 Properties of Clean Variable

Since clean flow ODE is a special case of clean flow SDE when $\beta_t = 0$, $\hat{\boldsymbol{x}}_t^{\mathrm{gt}}$ in the ODE also maintains the "clean properties" discussed in Appx. Sec. F.3.

H Gradient Variance

	VSD [44]	SDS [30]	FSD [48]	CFD (ours)
σ (↓)	5.165± 0.458	4.670±0.066	4.580±0.081	**4.521±0.090**

Table 6: **Scaled Gradient Variance.** Our CFD has the lowest gradient variance.

We compare the gradient variance of different method during training. We compute the scaled gradient variance by taking Exponential Moving Average parameters $\hat{v}_t$, $\hat{m}_t$ from Adam optimizer for

convenience. We report the scaled gradient variance σ on the parameters of nerf hash encoding with 10 seeds for each of the noising methods. σ was calculated according to (where g_t is the gradient):

$$\begin{cases} \hat{m}_t \approx \mathbb{E}[g_t], \\ \hat{v}_t \approx \mathbb{E}[g_t^2], \\ \sigma = \sqrt{\frac{\text{sum}(\hat{v}_t - \hat{m}_t^2)}{\text{sum}(\hat{v}_t)}} \approx \sqrt{\frac{\text{sum}(\text{Var}(g_t))}{\text{sum}(\hat{v}_t)}}. \end{cases} \tag{40}$$