Tortoise and Hare Guidance: Accelerating Diffusion Model Inference with Multirate Integration

Yunghee Lee **Byeonghyun Pak** **Junwha Hong** **Hoseong Kim**[†]
Agency for Defense Development
{yhl, byeonghyun_pak, qbit, hoseongkim}@add.re.kr

Abstract

In this paper, we propose **Tortoise and Hare Guidance (THG)**, a training-free strategy that accelerates diffusion sampling while maintaining high-fidelity generation. We demonstrate that the noise estimate and the additional guidance term exhibit markedly different sensitivity to numerical error by reformulating the classifier-free guidance (CFG) ODE as a *multirate system of ODEs*. Our error-bound analysis shows that the additional guidance branch is more robust to approximation, revealing substantial redundancy that conventional solvers fail to exploit. Building on this insight, THG significantly reduces the computation of the additional guidance: the noise estimate is integrated with the tortoise equation on the original, fine-grained timestep grid, while the additional guidance is integrated with the hare equation only on a coarse grid. We also introduce (i) an error-bound-aware timestep sampler that adaptively selects step sizes and (ii) a guidance-scale scheduler that stabilizes large extrapolation spans. THG reduces the number of function evaluations (NFE) by up to 30% with virtually no loss in generation fidelity (ΔImageReward ≤ 0.032) and outperforms state-of-the-art CFG-based training-free accelerators under identical computation budgets. Our findings highlight the potential of multirate formulations for diffusion solvers, paving the way for real-time high-quality image synthesis without any model retraining. The source code is available at `https://github.com/yhlee-add/THG`.

1 Introduction

Diffusion models (DMs) have become the state-of-the-art generative model for images [10, 40, 47] and, more recently, for video [20, 1, 52, 21] and audio-visual content [5, 41]. Despite their impressive quality, sampling is costly: each output is obtained by iteratively denoising a noisy sample, and the latency scales with the total number of function evaluations (NFE) required by the solver.

Many practical scenarios, such as text-to-image synthesis, class-controlled synthesis, or in-context image editing, require conditional generation. The dominant technique for high-quality conditioning is *classifier-free guidance* (CFG) [18], which improves perceptual quality and controllability. However, CFG runs the denoising network twice per timestep—once conditional and once unconditional—thereby doubling the NFE. For real-time applications, such as interactive editing and large-scale serving, evaluating a deep backbone at every timestep remains a major bottleneck.

A large body of work to accelerate these models has focused on two main approaches. Some approaches reduce the number of steps using higher-order ODE/SDE solvers [45, 46, 30] or distillation [43, 34], while others—such as cache-based strategies like DeepCache [33] and Learning-to-Cache [32]—lower the cost per step by reusing intermediate features. Nevertheless, both approaches still perform two forward passes whenever CFG is enabled, implicitly assuming that conditional and unconditional calls are equally indispensable.

[†]Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

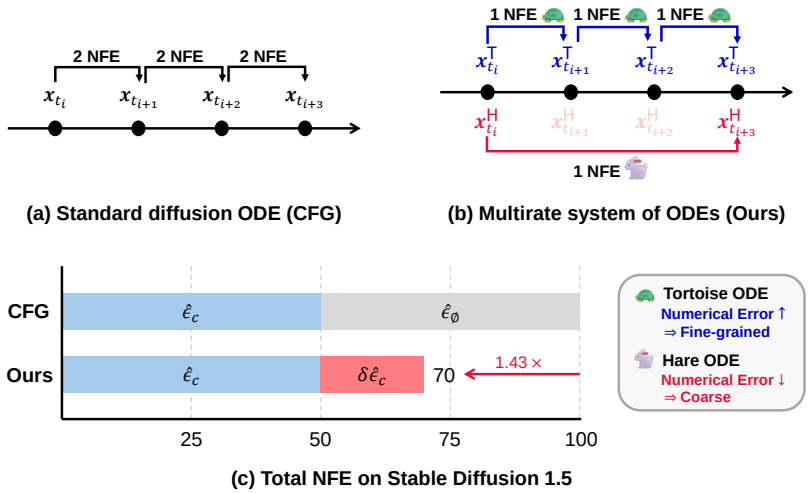


Figure 1: **Conceptual illustration of Tortoise and Hare Guidance.** We decompose the standard diffusion ODE into a tortoise branch (Eq. 6), which is numerically sensitive and thus integrated on a fine-grained grid, and a hare branch (Eq. 7), which is comparatively less sensitive and can be integrated with larger step sizes. Our multirate scheme evaluates each branch at different timestep grids, skipping unnecessary evaluations, thereby boosting inference efficiency without sacrificing sample quality.

Through the lens of numerical analysis, we revisit CFG by reformulating the reverse diffusion process as a two-state multirate system of ODEs whose trajectories are governed by the noise estimate and the additional guidance term. Our error-bound analysis reveals a pronounced asymmetry: the additional guidance term is more robust to approximation than the noise estimate, exposing substantial redundancy that conventional solvers fail to exploit. This finding raises a natural question: *Do we need to compute the neural network twice at every fine-grained timestep?*

Leveraging this asymmetry, we introduce **Tortoise and Hare Guidance (THG)**, a training-free sampler that bypasses most additional guidance computation. The noise estimate is integrated with the tortoise equation on the original fine-grained timestep grid. Meanwhile, the additional guidance is integrated with the hare equation only on a coarse grid. We further introduce (i) an error-bound-aware timestep sampler that adaptively determines the coarse grid, and (ii) a guidance-scale scheduler that keeps the trajectory stable over significant gaps.

With these components, THG achieves sampling speeds up to $1.43\times$ faster by reducing the NFE budget from 100 to as low as 70 while maintaining virtually identical generation fidelity (ΔImageReward ≤ 0.032). Moreover, across Stable Diffusion 1.5 [40], 3.5 Large [47], and AudioLDM 2 [28], our method outperforms state-of-the-art CFG-based training-free accelerators under identical computation budgets. Our study highlights the potential of multirate formulations for accelerating diffusion models and brings us a step closer to achieving real-time performance and high-quality image synthesis without retraining the model.

In summary, our contributions are threefold:

- We are the first to cast the reverse diffusion ODE as a two-state multirate system of ODEs and to provide an error-bound analysis showing that the additional guidance term can be safely approximated at a much coarser temporal resolution.
- We design **Tortoise and Hare Guidance (THG)**, a training-free sampler that eliminates the need for a significant amount of additional guidance term evaluation. THG is compatible with any diffusion backbone.
- Using image-text pairs from the COCO 2014 dataset, we demonstrate that THG can reduce NFEs up to 30% with virtually no loss in generation fidelity (ΔImageReward ≤ 0.032). THG outperforms state-of-the-art CFG-based accelerators under identical compute budgets.

2 Related work

Diffusion models Denoising Diffusion Probabilistic Models (DDPMs) [19] laid the foundation for modern diffusion models by introducing a probabilistic framework. A forward Markov process gradually corrupts a data point x_0 into Gaussian noise. In the reverse process, at each timestep t, a neural network $\hat{\epsilon}_\theta(x_t, t)$ estimates and removes the noise component in x_t to recover x_{t-1}, ultimately reconstructing x_0. The denoising trajectory can be interpreted either as a stochastic differential equation (SDE) or its deterministic counterpart, the probability flow ODE (PF-ODE) [46]. Denoising Diffusion Implicit Models (DDIMs) [45] drop the strict Markov assumption of DDPMs and apply Tweedie's formula [9] to jump directly from x_t to x_s, cutting sampling steps from hundreds of steps to as few as 50 and effectively solving the PF-ODE in a single deterministic pass [46].

ODE-based integrators Viewing diffusion sampling as an initial-value ODE problem enables high-order integration techniques. Concretely, DPM-solver [30] observes that the diffusion ODE

$$\mathrm{d}x_t/\mathrm{d}t = f(t)x_t + (g^2(t)/2\sigma_t)\hat{\epsilon}_\theta(x_t) \tag{1}$$

has a semi-linear term $f(t)x_t$. The need for approximation for the linear term is eliminated by solving the semi-linear ODE using the *variation of constants* formula. This semi-linear integrator then affords large step sizes with minimal approximation error. Inspired by these semi-linear methods, we introduce a multirate formulation for the classifier-free guidance (CFG) scheme [18] that adjusts the step size of each component of CFG to its own dynamics, achieving further reductions in the number of function evaluations (NFE) without degrading sample quality.

Classifier-free guidance and its variations In real-world applications, diffusion models must produce samples that satisfy a given condition (e.g., class label or text prompt). Classifier Guidance [8] achieves this by incorporating a pre-trained classifier $p_\phi(c|x_t)$, effectively sampling from the *sharpened* density $p(x)p(c|x)^\omega$, where ω controls the strength of the bias towards class c. Classifier-Free Guidance (CFG) [18] eliminates the need for an external classifier by training a single denoising network that gives both conditional and unconditional outputs. Concretely, if $\hat{\epsilon}_\theta(x_t, c)$ and $\hat{\epsilon}_\theta(x_t, \varnothing)$ denote the network's noise predictions with and without condition c, respectively, then CFG defines

$$\hat{\epsilon}^{\text{CFG}}_\theta(x_t, c) = \hat{\epsilon}_\theta(x_t, \varnothing) + \omega \cdot (\hat{\epsilon}_\theta(x_t, c) - \hat{\epsilon}_\theta(x_t, \varnothing)). \tag{2}$$

Subsequent variants focus on finding the optimal strength and timing of guidance for balancing condition fidelity against sample diversity. Guidance Interval [26] restricts the use of CFG to mid-level noise steps, avoiding over-conditioning at the beginning and final stages of the sampling process. CADS and Dynamic-CFG [42] slowly anneal either the conditioning vector or the scale ω during the early denoising steps, preserving diversity in the final samples. PCG [2] reformulates CFG as a predictor-corrector method (with $\omega' = 2\omega - 1$) that alternates between denoising and sharpening phases. CFG++ [7] treats guidance as an explicit loss term rather than a sampling bias, splitting each DDIM iteration into "denoising" and "renoising" phases. Unlike these methods, we reformulate the diffusion ODE using a multirate method, integrating the noise estimate on a fine-grained grid and the additional guidance term on a coarse grid, reducing the NFE while preserving sample quality.

Efficient diffusion models Beyond advanced ODE/SDE solvers, various methods have been proposed to speed up pre-trained diffusion models. Distillation methods [43, 34] compress a pre-trained "teacher" model into a "student" model that can advance multiple timesteps in one forward pass. While these methods reduce the number of sampling steps, they incur substantial retraining costs. Cache-based techniques exploit feature redundancy within the denoising neural network $\hat{\epsilon}_\theta$. DeepCache [33] reuses high-level U-Net activations across adjacent steps. Learning-to-Cache [32] introduces a layer-wise caching mechanism that dynamically reuses transformer activations across timesteps via a timestep-conditioned router. Δ-Dit [4] leverages stage-adaptive caching of block-specific feature offsets in DiT models to speed up inference without retraining. These methods deliver inference speedups without retraining but depend heavily on the model's internal architecture. More recently, several works have noted that CFG doubles the NFE per denoising step and have proposed methods to reduce this extra cost. Adaptive Guidance [3] adaptively skips redundant guidance steps based on cosine similarity between conditional and unconditional predictions. FasterCache [31] reuses attention features and conditional-unconditional residuals to mitigate CFG overhead. Although these methods reduce the NFE, they lack a rigorous theoretical foundation and leave further savings on the table. Our approach delivers a more efficient and theoretically grounded method of guided diffusion by directly exploiting the CFG's intrinsic dynamics.

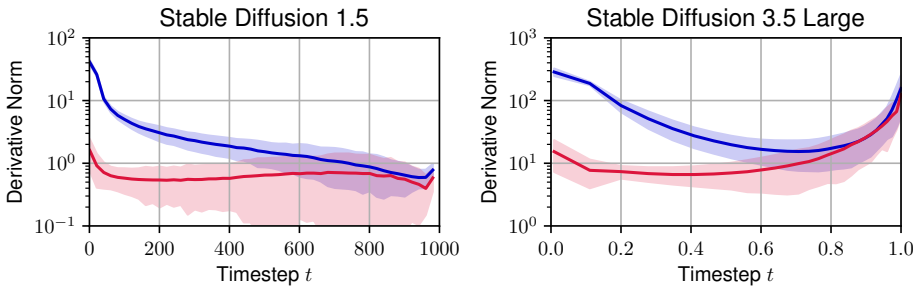


Figure 2: **Time-derivative norms of the noise estimate $\hat{\epsilon}_c(x_t)$ and additional guidance $\Delta\hat{\epsilon}_c(x_t)$.** We plot the L2 norms of the time derivatives $\frac{\mathrm{d}}{\mathrm{d}t}\hat{\epsilon}_c(x_t)$ and $\frac{\mathrm{d}}{\mathrm{d}t}\Delta\hat{\epsilon}_c(x_t)$ across diffusion timesteps for Stable Diffusion 1.5 and 3.5 Large. The results confirm that the noise estimate exhibits greater temporal sensitivity compared to the guidance term. Shaded areas denote two standard deviations over multiple prompts.

3 Method

In this section, we introduce **Tortoise and Hare Guidance (THG)**, which accelerates diffusion model inference by leveraging the asymmetry between the noise estimate and the additional guidance terms. Since the additional guidance term varies more slowly *w.r.t.* the denoising timestep t than the noise estimate term, we apply a multirate integration scheme that uses a coarser timestep grid for the additional guidance term (Sec. 3.1 and Sec. 3.2). We then perform an approximation error-bound analysis to determine the appropriate grid granularity (Sec. 3.3). Finally, we propose an adaptive guidance scale to compensate for any performance degradation resulting from the reduced number of evaluation points (Sec. 3.4).

Preliminaries To accommodate different definitions of the diffusion process [19, 46, 49], we adopt a general notation [30] so that the forward process and the diffusion ODE are described as follows:

$$q(x_t|x_0) := \mathcal{N}(x_t; \alpha_t x_0, \sigma_t^2 I), \quad \frac{\mathrm{d}x_t}{\mathrm{d}t} = f(t)x_t + \frac{g^2(t)}{2\sigma_t}\hat{\epsilon}_\theta(x_t), \quad x_T \sim \mathcal{N}(0, \sigma_T^2 I), \tag{3}$$

where $f(t) = \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}$, $g^2(t) = \frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\sigma_t^2$, and $t \in [0, T]$. (v-prediction models are covered in Appendix A.) α_t and σ_t are the predefined noise schedule of the diffusion model. Although modern diffusion models primarily operate in the latent space [40], we adopt x (instead of z), as our framework is agnostic to this choice. For brevity, we denote the unconditional noise estimate $\hat{\epsilon}_\varnothing(x_t) := \hat{\epsilon}_\theta(x_t, \varnothing)$, the conditional noise estimate $\hat{\epsilon}_c(x_t) = \hat{\epsilon}_\theta(x_t, c)$, the difference of the two $\Delta\hat{\epsilon}_c(x_t) := \hat{\epsilon}_c(x_t) - \hat{\epsilon}_\varnothing(x_t)$, and the CFG noise estimate $\hat{\epsilon}_c^\omega(x_t) = \hat{\epsilon}_\theta^{\text{CFG}}(x_t, c)$ following [7].

3.1 A multirate formulation

We propose a multirate formulation [39], in which the reverse diffusion process is decomposed into numerically sensitive and less sensitive components to reduce the number of function evaluations (NFE). We begin by writing the diffusion ODE in Eq. 3 by explicitly separating it into two distinct terms, the noise estimate and the additional guidance term. By the definition of CFG, we have

$$\hat{\epsilon}_\theta(x_t) := \hat{\epsilon}_c^\omega(x_t) = \hat{\epsilon}_\varnothing(x_t) + \omega \cdot \Delta\hat{\epsilon}_c(x_t) \equiv \hat{\epsilon}_c(x_t) + (\omega - 1) \cdot \Delta\hat{\epsilon}_c(x_t). \tag{4}$$

Substituting Eq. 4 into Eq. 3 yields the following:

$$\frac{\mathrm{d}}{\mathrm{d}t}x_t = f(t)x_t + \frac{g^2(t)}{2\sigma_t}\hat{\epsilon}_c^\omega(x_t) = f(t)x_t + \underbrace{\frac{g^2(t)}{2\sigma_t}\hat{\epsilon}_c(x_t)}_{\text{sensitive}} + \underbrace{\frac{g^2(t)}{2\sigma_t}(\omega - 1)\Delta\hat{\epsilon}_c(x_t)}_{\text{less sensitive}}. \tag{5}$$

We observe a significant difference in temporal sensitivity between the noise estimate term and the additional guidance term. Figure 2 plots the time-derivative norms of $\hat{\epsilon}_c(x_t)$ and $\delta\hat{\epsilon}_c(x_t)$, confirming that the noise estimate varies more rapidly than the additional guidance term. This result

Algorithm 1 Tortoise and Hare Guidance Algorithm

Require: $x_T \sim \mathcal{N}(0, \sigma_T^2 I)$ ▷ Initial noise
Require: $\omega \geq 0$ ▷ Guidance scale
Require: $\{t_i\}_{0 \leq i \leq N}, t_0 = T, t_N = 0$ ▷ Fine-grained timestep grid
Require: $C \subset \{t_i | 0 \leq i \leq N\}, 0 \in C, T \in C$ ▷ Coarse timestep grid

$x_T^{\mathsf{T}} \leftarrow x_T$
$x_T^{\mathsf{H}} \leftarrow 0$
for $i = 0$ **to** $N - 1$ **do**
  $\hat{\epsilon}_c \leftarrow \hat{\epsilon}_\theta(x_{t_i}^{\mathsf{T}} + x_{t_i}^{\mathsf{H}}, c)$ ▷ 1 NFE
  $x_{t_{i+1}}^{\mathsf{T}} \leftarrow \text{Solver}(x_{t_i}^{\mathsf{T}}, \hat{\epsilon}_c, t_i, t_{i+1})$ ▷ Compute $x_{t_{i+1}}^{\mathsf{T}}$ given $x_{t_i}^{\mathsf{T}}$
  if $t_i \in C$ **then**
    $\hat{\epsilon}_\varnothing \leftarrow \hat{\epsilon}_\theta(x_{t_i}^{\mathsf{T}} + x_{t_i}^{\mathsf{H}}, \varnothing)$ ▷ 1 NFE (only if $t_i \in C$)
    $\Delta\hat{\epsilon}_c \leftarrow \hat{\epsilon}_c - \hat{\epsilon}_\varnothing$
    $j \leftarrow i$
    repeat ▷ Compute x^{H} up to the next coarse timestep
      $j \leftarrow j + 1$
      $x_{t_j}^{\mathsf{H}} \leftarrow \text{Solver}(x_{t_i}^{\mathsf{H}}, (\omega - 1) \cdot \Delta\hat{\epsilon}_c, t_i, t_j)$ ▷ Compute $x_{t_j}^{\mathsf{H}}$ given $x_{t_i}^{\mathsf{H}}$
    until $t_j \in C$ ▷ t_j equals the next coarse timestep at inner loop exit
  end if
end for
$x_0 \leftarrow x_0^{\mathsf{T}} + x_0^{\mathsf{H}}$
return x_0

clearly demonstrates that the noise estimate exhibits greater numerical sensitivity than the additional guidance.

This motivates the use of a multirate method [44] where the sensitive term is integrated on a fine-grained grid, and the less sensitive term is integrated on a coarse grid. We split the diffusion ODE (Eq. 5) into the following system of ODEs:

$$\frac{\mathrm{d}}{\mathrm{d}t} x_t^{\mathsf{T}} = f(t) x_t^{\mathsf{T}} + \frac{g^2(t)}{2\sigma_t} \hat{\epsilon}_c(x_t^{\mathsf{T}} + x_t^{\mathsf{H}}), \tag{6}$$

$$\frac{\mathrm{d}}{\mathrm{d}t} x_t^{\mathsf{H}} = f(t) x_t^{\mathsf{H}} + \frac{g^2(t)}{2\sigma_t} (\omega - 1) \Delta\hat{\epsilon}_c(x_t^{\mathsf{T}} + x_t^{\mathsf{H}}), \tag{7}$$

where $x_T^{\mathsf{T}} = x_T$, $x_T^{\mathsf{H}} = 0$, and $x_t := x_t^{\mathsf{T}} + x_t^{\mathsf{H}}$. The tortoise x_t^{T} covers the noise estimate part of the diffusion ODE, while the hare x_t^{H} takes care of the additional guidance term. We call the ODE integrated on the fine-grained grid the tortoise equation (Eq. 6), and the ODE integrated on the coarse grid the hare equation (Eq. 7). Intuitively, the hare equation uses coarser timestep intervals—i.e. larger steps—allowing it to skip unnecessary computation and thus significantly improve the efficiency of integrating the diffusion ODE. Moreover, because both equations retain the standard diffusion ODE form, existing solvers such as DDIM [45] can be applied to each equation without modification.

3.2 Tortoise and Hare Guidance

Solving the hare equation (Eq. 7) on the coarse grid is straightforward, since every coarse timestep is also a fine-grained timestep. By contrast, because the tortoise equation (Eq. 6) requires the full state $x_t = x_t^{\mathsf{T}} + x_t^{\mathsf{H}}$ at every fine-grained timestep, we must infer x_t^{H} at those intermediate points [39]. Instead of using generic extrapolation methods [31], we exploit a property of diffusion model solvers: given x_t and $\hat{\epsilon}_\theta(x_t)$, they can deterministically compute x_s for any $s < t$ by running the chosen solver from t to s. From each coarse timestep, we run the solver not only to compute x_t^{H} for the next coarse timestep but also to populate x_t^{H} for all intermediate fine-grained timesteps, thereby constructing the full trajectory of x_t^{H} on the fine-grained grid for use in integrating the tortoise equation.

Building on this formulation, we propose an implementation strategy summarized in Algorithm 1. While the standard diffusion solver evaluates both $\hat{\epsilon}_c(x_t)$ and $\Delta\hat{\epsilon}_c(x_t)$ at every fine-grained timestep, our scheme evaluates $\Delta\hat{\epsilon}_c(x_t)$ only on the coarse grid $C \subset \{t_0, \ldots, t_N\}$, thereby significantly reducing NFE. At each coarse step $t_i \in C$, the updated guidance term is used to integrate the hare

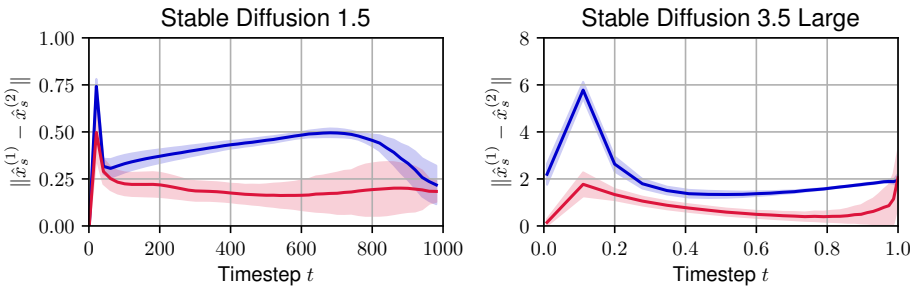


Figure 3: **Approximation error bounds of the tortoise x_t^{T} and the hare x_t^{H}.** We show the per-timestep error bound of the tortoise and the hare terms across sampling steps. The consistently higher bounds for the tortoise curve indicate that the noise estimate is more sensitive to timestep resolution than the additional guidance. Shaded areas denote two standard deviations over multiple prompts.

equation across the fine-grained grid until the next coarse step. We then use the resulting x_t^{H} values during the subsequent tortoise equation steps. As a result, the NFE is reduced from $2N$ to $N + |C| - 1$ while preserving the dynamics of the original diffusion ODE. Moreover, it slots seamlessly into existing diffusion pipelines without any changes to their core logic.

3.3 Approximation error bound analysis

To determine an appropriate coarse grid C for the hare equation, we now turn to an error-based criterion. Our objective is to ensure that the integration error of x_t^{H} remains sufficiently small relative to that of x_t^{T}. To this end, we adopt a standard multirate strategy [11]. We select coarse step sizes such that the ratio between the hare's approximation error and the tortoise's approximation error does not exceed a user-specified threshold ρ such that $\rho \approx 1$:

$$\frac{\|\hat{x}_s^{\mathsf{H}} - x_s^{\mathsf{H}}\|}{\|\hat{x}_s^{\mathsf{T}} - x_s^{\mathsf{T}}\|} \leq \rho. \tag{8}$$

Here, x_s^{T} and x_s^{H} denote the analytical solutions to the tortoise and hare equations at timestep s, while $\hat{x}_s^{\mathsf{T}}$ and $\hat{x}_s^{\mathsf{H}}$ are the corresponding numerical solutions obtained using the diffusion model solver. Given that the solver has order p, the local integration error at a single step scales as [14]:

$$\hat{x}_s - x_s = c \cdot (\Delta t)^{p+1} + \mathcal{O}((\Delta t)^{p+2}) \tag{9}$$

where Δt is the fine-grained step size and c is an unknown constant. Let the coarse step size be $m\Delta t$, meaning the hare *leaps* m tortoise steps per update. Then, the local integration error of the hare equation over one coarse step becomes:

$$\hat{x}_s^{\mathsf{H}} - x_s^{\mathsf{H}} = c^{\mathsf{H}} \cdot (m\Delta t)^{p+1} + \mathcal{O}\left((\Delta t)^{p+2}\right). \tag{10}$$

In contrast, the tortoise equation accumulates error over m fine-grained steps:

$$\hat{x}_s^{\mathsf{T}} - x_s^{\mathsf{T}} = c^{\mathsf{T}} \cdot m(\Delta t)^{p+1} + \mathcal{O}\left((\Delta t)^{p+2}\right), \tag{11}$$

Taking the ratio from Eq. 8 and ignoring higher-order terms, we obtain:

$$\frac{\|\hat{x}_s^{\mathsf{H}} - x_s^{\mathsf{H}}\|}{\|\hat{x}_s^{\mathsf{T}} - x_s^{\mathsf{T}}\|} = \frac{\|c^{\mathsf{H}}\| m^{p+1} (\Delta t)^{p+1}}{\|c^{\mathsf{T}}\| m (\Delta t)^{p+1}} = m^p \frac{\|c^{\mathsf{H}}\|}{\|c^{\mathsf{T}}\|} \leq \rho, \quad \therefore m \leq \left(\rho \|c^{\mathsf{T}}\| / \|c^{\mathsf{H}}\|\right)^{1/p}. \tag{12}$$

Since m must be a positive integer, we define the maximum allowable value as:

$$m_{\max} := \max\left(1, \left\lfloor \left(\rho \|c^{\mathsf{T}}\| / \|c^{\mathsf{H}}\|\right)^{1/p} \right\rfloor\right). \tag{13}$$

Estimating the error constants To compute $m_{\max}$, we need estimates of $\|c^{\mathsf{T}}\|$ and $\|c^{\mathsf{H}}\|$ without relying on the analytic solution x_s. We accomplish this using the Richardson extrapolation method [14] . First, solve the ODE once using step size Δt:

$$\hat{x}_s^{(1)} - x_s = c \cdot (\Delta t)^{p+1} + \mathcal{O}\left((\Delta t)^{p+2}\right). \tag{14}$$

Algorithm 2 Look before you leap

Require: $m_{\max}(t_i)$ ▷ Calculated $m_{\max}$ for each timestep
Require: $\{t_i\}_{0 \le i \le N}, t_0 = T, t_N = 0$ ▷ Fine-grained timestep grid
$C \leftarrow \{\}$ ▷ The result is initially an empty set
$i \leftarrow 0$ ▷ Start advancing the fine-grained grid from the first timestep
while $i < N$ **do**
  $C \leftarrow C \cup \{t_i\}$ ▷ Add current position
  $i \leftarrow i + m_{\max}(t_i)$ ▷ Advance $m_{\max}(t_i)$ steps
end while
$C \leftarrow C \cup \{0\}$ ▷ Include last timestep
return C

Next, solve again using two steps of size $\Delta t/2$:

$$\hat{x}_s^{(2)} - x_s = c \cdot 2(\Delta t/2)^{p+1} + \mathcal{O}\left((\Delta t)^{p+2}\right). \tag{15}$$

Subtracting Eq. 14 and Eq. 15 yields

$$\hat{x}_s^{(1)} - \hat{x}_s^{(2)} = c \cdot \left(1 - 2^{-p}\right)(\Delta t)^{p+1} + \mathcal{O}\left((\Delta t)^{p+2}\right). \tag{16}$$

If we ignore the higher-order terms, the norm of this difference provides a direct estimate proportional to $\|c\|$. We apply this procedure independently to both the tortoise and hare equations to estimate $\|c^{\mathsf{T}}\|$ and $\|c^{\mathsf{H}}\|$, respectively. Empirical results (Fig. 3) on 30,000 prompts from the COCO 2014 dataset [27, 37] show that $\|c^{\mathsf{T}}\|$ is greater than $\|c^{\mathsf{H}}\|$ for most cases, confirming that the tortoise equation is more sensitive to timestep resolution.

Example usage scenario To generate samples using Tortoise and Hare Guidance, we first calculate $m_{\max}$ of each fine-grained timestep (Eq. 13) using the average $\|c^{\mathsf{T}}\|$ and $\|c^{\mathsf{H}}\|$ over a batch of inputs. (More details are covered in Appendix C.) Then we build the coarse timestep grid C via the "look before you leap" strategy (Algorithm 2). Starting at the first fine-grained timestep t_0, we insert coarse timesteps so that they lie $m_{\max}(t_i)$ steps ahead, keeping the local error ratio below ρ. Finally, we generate samples using Algorithm 1. Note that C could be reused for all subsequent inferences without any additional NFEs.

3.4 Adjusting Guidance Scales

Approximating the hare at fine-grained timesteps can lead to a degradation in output quality. To compensate for this, we propose adjusting the guidance scale whenever the additional guidance term is used more than once per timestep. In particular, we introduce a constant boost factor b and scale the guidance term: $\Delta\hat{\epsilon}_c \leftarrow b \cdot \Delta\hat{\epsilon}_c$. This simple multiplicative adjustment improves sample quality, especially in cases where the inner loop (which integrates the hare equation) is repeated multiple times for each coarse step. Our method draws inspiration from prior work such as CFG-Cache [31], which amplifies guidance in the frequency domain using FFT. However, unlike FFT-based methods, our approach avoids the overhead of spectral transforms, which can be computationally expensive for high-dimensional latent variables. The additional guidance term predominantly contains low-frequency information in the early stages of sampling and vice versa [15]. Therefore, selectively enhancing the frequency components of the additional guidance term per timestep has low significance.

Furthermore, CFG and the additional guidance term are of low significance at the later phase of the reverse diffusion process [26, 3]. We leverage this fact by introducing a threshold timestep index i_{hi} and substituting $\Delta\hat{\epsilon}_c \leftarrow 0$ if $i \ge i_{\text{hi}}$. This simple adjustment helps reduce the NFE even further.

4 Experiments

4.1 Experimental Settings

Compared methods To demonstrate the effectiveness of our approach, we compare against CFG-Cache [31], a training-free acceleration technique that reuses conditional and unconditional outputs in video diffusion models. Given that CFG-Cache exploits a timestep-adaptive enhancement technique

Table 1: Comparison of methods in terms of distributional similarity and prompt fidelity. Our method is marked in blue, whereas vanilla CFG is marked in gray. The best results are **highlighted**.

Method	NFE ↓	Distributional similarity FID ↓	Distributional similarity CMMD ↓	Prompt fidelity CS ↑	Prompt fidelity IR ↑
Stable Diffusion 1.5 with DDIM					
CFG [18]	100	14.057	0.58885	26.294	0.14765
CFG-Cache w/o FFT [31]	70	14.240	**0.59187**	26.141	0.08757
CFG-Cache [31]	70	14.367	0.59556	26.180	0.09735
THG (Ours)	70	**14.165**	0.59223	**26.189**	**0.11499**
Stable Diffusion 1.5 with DPM-Solver-2 [30]					
CFG [18]	100	13.255	0.60379	26.254	0.16148
CFG-Cache w/o FFT [31]	70	13.387	**0.60665**	26.107	0.10513
CFG-Cache [31]	70	13.468	0.60880	26.147	0.11474
THG (Ours)	70	**12.909**	0.60868	**26.205**	**0.14926**
Stable Diffusion 1.5 with 2nd-order Linear Multistep Method [29]					
CFG [18]	100	13.540	0.60653	26.260	0.15966
CFG-Cache w/o FFT [31]	70	**13.686**	**0.60844**	26.107	0.09881
CFG-Cache [31]	70	13.798	0.61142	26.144	0.10805
THG (Ours)	70	**13.686**	0.61094	**26.204**	**0.15184**
Stable Diffusion 3.5 Large with Euler method					
CFG [18]	56	68.158	0.81106	26.624	1.03569
CFG-Cache w/o FFT [31]	38	67.931	0.76448	26.643	1.00715
CFG-Cache [31]	38	**67.914**	**0.75324**	26.668	1.00745
THG (Ours)	38	68.252	0.80092	**26.672**	**1.02365**

Method	NFE ↓	Distributional similarity FAD ↓	Prompt fidelity CLAP Score ↑
AudioLDM 2 with DDIM			
CFG [18]	100	2.596	0.2409
CFG-Cache w/o FFT [31]	70	2.901	0.2251
THG (Ours)	70	**2.764**	**0.2342**

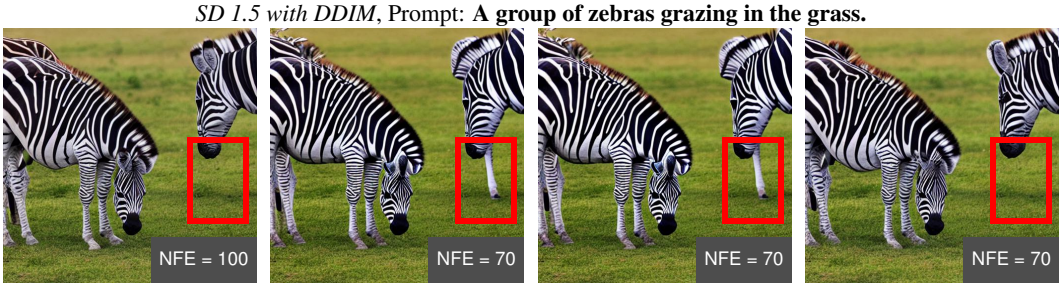
to mitigate fine-detail degradation, we evaluate both the full CFG-Cache (with enhancement) and a variant without this enhancement (denoted "CFG-Cache w/o FFT"). All variants are adapted to the diffusion model's modality for a fair comparison. More comparisons with CFG variants are given in Appendix E.

Implementation details We build Tortoise and Hare Guidance with PyTorch [36], Diffusers [48], and Accelerate [13]. We evaluate three pretrained diffusion models—Stable Diffusion 1.5 [40], Stable Diffusion 3.5 Large [47, 10], and AudioLDM 2 [28]. For Stable Diffusion (SD) models, we use prompt–image pairs randomly sampled from COCO 2014 [27, 37]: 30,000 pairs for SD 1.5 and 1,000 pairs for SD 3.5 Large. For AudioLDM 2, we use 2,230 prompt-audio pairs from the validation set of AudioCaps [25]. Experiments are run on a server with an AMD EPYC 74F3 26934-core CPU, 1 TB of RAM, and 8 NVIDIA A100 80GB GPUs. Hyperparameters $(N, \omega, \rho, b, i_{\text{hi}})$ are set to $(50, 7.5, 1.1, 1.1, 38)$ for SD 1.5, $(28, 3.5, 1.0, 1.2, 21)$ for SD 3.5 Large, and $(50, 3.5, 0.9, 1.15, 39)$ for AudioLDM 2.

4.2 Main Results

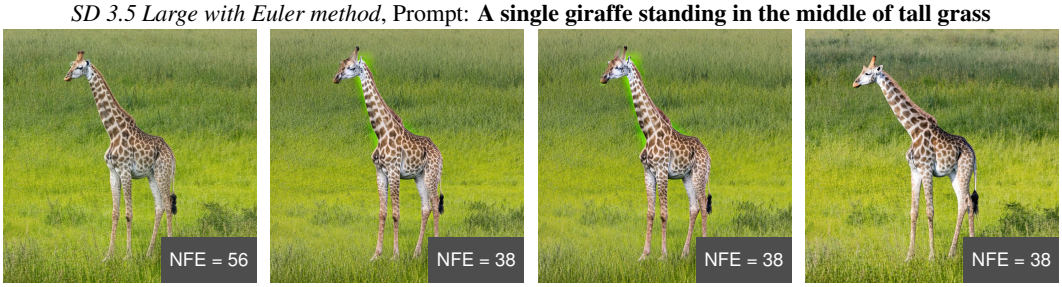
Quantitative comparison Table 1 compares our method to the CFG-Cache variants in terms of distributional similarity metrics such as FID [17, 35], CMMD [22], and FAD [24], together with prompt fidelity metrics such as CLIP Score (CS) [16], ImageReward (IR) [51], and CLAP Score [50] under the same number of function evaluations (NFE). Refer to Appendix F for more details on metrics. On SD 1.5, all methods cut NFE from 100 to 70; ours lowers FID (14.165 vs. 14.240),

SD 1.5 with DDIM, Prompt: **A group of zebras grazing in the grass.**

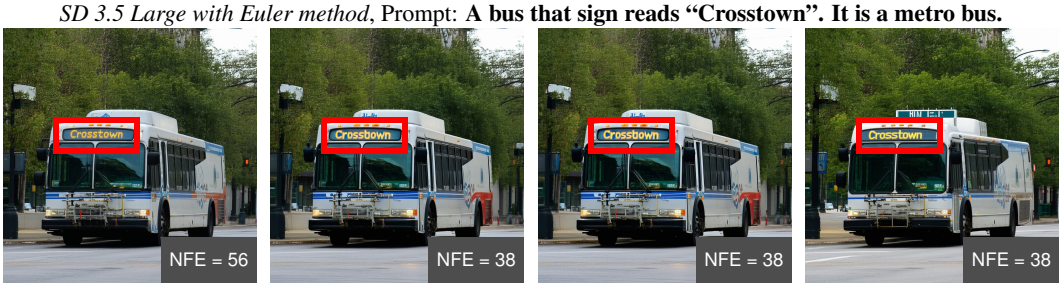


SD 1.5 with DDIM, Prompt: **Two cows on a hill above a valley and mountains on the other side.**

SD 3.5 Large with Euler method, Prompt: **A single giraffe standing in the middle of tall grass**



SD 3.5 Large with Euler method, Prompt: **A bus that sign reads "Crosstown". It is a metro bus.**



(a) CFG (b) CFG-Cache w/o FFT (c) CFG-Cache (d) THG (Ours)

Figure 4: **Comparison of visual results** for prompts from the COCO 2014 dataset.

matches CMMD, and improves CS and IR over CFG-Cache w/o FFT, and beats full CFG-Cache on CS and IR while keeping FID competitive. On SD 3.5 Large, all cut NFE from 56 to 38; although CFG-Cache slightly leads on FID and CMMD, our method delivers nearly equal FID/CMMD with the highest IR and tied CS. On AudioLDM 2 [28], all cut NFE from 100 to 70; ours lowers FAD (2.764 vs. 2.901) and improves CLAP Score over CFG-Cache w/o FFT. CFG-Cache is excluded since its enhancement is inapplicable to the audio domain. These results show that THG generalizes across solvers and scales, preserving sample distribution and text alignment under aggressive step reduction. The tradeoff of distributional similarity and prompt fidelity is further discussed in Appendix G.

Qualitative comparison Figure 4 compares images generated by our method and the two CFG-Cache variants. The results demonstrate that THG effectively preserves image fidelity and fine details. More visual comparisons are shown in Appendix I.

Table 2: Ablation study for the hyperparameter b.

Method	NFE ↓	FID ↓	CMMD ↓	CS ↑	IR ↑
$b = 1.00$	70	13.811	0.58364	26.137	0.09395
$b = 1.05$	70	13.988	0.58794	26.162	0.10456
$b = 1.10$	70	14.232	0.59354	26.197	0.11576
$b = 1.15$	70	14.472	0.59783	26.221	0.12639
$b = 1.20$	70	14.729	0.60260	26.246	0.13478

Table 3: Ablation study for the hyperparameter ρ.

Method	NFE ↓	FID ↓	CMMD ↓	CS ↑	IR ↑
$\rho = 0.9$	75	14.128	0.59044	26.193	0.11942
$\rho = 1.0$	73	14.148	0.59068	26.200	0.11949
$\rho = 1.1$	70	14.232	0.59354	26.197	0.11576
$\rho = 1.2$	69	14.336	0.59306	26.221	0.11262
$\rho = 1.3$	67	14.280	0.59521	26.197	0.10849

4.3 Ablation Studies

Boost factor b We perform ablation studies on hyperparameters using SD 1.5 with DDIM. Table 2 shows how varying the boost factor b affects inference quality at 70 NFE budget with the same latents x_T. As b increases from 1.00 to 1.20, we observe a steady rise in IR from 0.09395 up to 0.13478, indicating stronger image–text alignment, and a modest gain in CS. However, this comes at the cost of higher FID and CMMD values, reflecting a gradual drop in distributional similarity. We select $b = 1.10$ as our default because it strikes the best balance: it substantially boosts IR (0.11576) with only a moderate increase in FID (14.232) and CMMD (0.59354) relative to lower b values.

Error-ratio threshold ρ Table 3 summarizes the effect of varying ρ with the same latents x_T. Lowering ρ from 1.1 to 0.9 results in more conservative hare leaps—NFE rise from 70 to 75—and yields slightly better FID (14.128 vs. 14.232) and CMMD (0.59044 vs. 0.59354), at the expense of marginally lower IR (0.11942 vs. 0.11576). Increasing ρ to 1.3 reduces NFE to 67 but degrades FID (14.280) and IR (0.10849). We choose $\rho = 1.1$ as our default since it achieves the best trade-off: a 30% NFE reduction (70 NFE) while maintaining competitive fidelity and alignment metrics.

5 Conclusion

We present Tortoise and Hare Guidance, a training-free acceleration framework for diffusion sampling that leverages a multirate reformulation of classifier-free guidance (CFG). Exploiting the asymmetric sensitivity of the noise estimate and the additional guidance term to numerical error, Tortoise and Hare Guidance integrates the noise estimate on a fine-grained grid while integrating the additional guidance term on a coarse grid. This approach allows for a substantial reduction in the number of function evaluations (NFE) without sacrificing generation quality. With an error-bound-aware timestep sampler and a guidance scale adjustment, our method achieves up to 30% faster sampling while preserving fidelity across models like Stable Diffusion 1.5, 3.5 Large, and AudioLDM 2, demonstrating the effectiveness of multirate integration for real-time high-quality generation.

Limitations Our experiments are currently limited to latent diffusion models and a few benchmark datasets such as COCO 2014 and AudioCaps. Extending the evaluation to a wider range of architectures, modalities, and downstream tasks will help assess the generality and robustness of our method.

Broader Impact By reducing sampling cost without retraining, Tortoise and Hare Guidance lowers the barrier to deploying diffusion models in real-time applications such as creative tools, accessibility services, and mobile environments. This could result in accelerating the production of synthetic media, including deepfakes and misleading content. Nonetheless, the capabilities of Tortoise and Hare Guidance remain bounded by those of the underlying diffusion model, introducing a limited impact to the quality of such synthetic media.

Acknowledgments and Disclosure of Funding

We sincerely thank Byeongju Woo, Chanyong Lee, and Eunjin Koh for their constructive discussions and support. We also appreciate Minseo Kim and Minkyu Song for providing insightful feedback. This work was supported by the Agency For Defense Development Grant Funded by the Korean Government (912A45701).

References

[1] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

[2] Arwen Bradley and Preetum Nakkiran. Classifier-free guidance is a predictor-corrector. *arXiv preprint arXiv:2408.09000*, 2024.

[3] Angela Castillo, Jonas Kohler, Juan C Pérez, Juan Pablo Pérez, Albert Pumarola, Bernard Ghanem, Pablo Arbeláez, and Ali Thabet. Adaptive guidance: Training-free acceleration of conditional diffusion models. *arXiv preprint arXiv:2312.12487*, 2023.

[4] Pengtao Chen, Mingzhu Shen, Peng Ye, Jianjian Cao, Chongjun Tu, Christos-Savvas Bouganis, Yiren Zhao, and Tao Chen. Δ-DiT: A Training-free Acceleration Method Tailored for Diffusion Transformers. *arXiv preprint arXiv:2406.01125*, 2024.

[5] Ho Kei Cheng, Masato Ishii, Akio Hayakawa, Takashi Shibuya, Alexander Schwing, and Yuki Mitsufuji. Taming multimodal joint training for high-quality video-to-audio synthesis. *arXiv preprint arXiv:2412.15322*, 2024.

[6] Jinyoung Choi, Junoh Kang, and Bohyung Han. Enhanced diffusion sampling via extrapolation with multiple ode solutions. *arXiv preprint arXiv:2504.01855*, 2025.

[7] Hyungjin Chung, Jeongsol Kim, Geon Yeong Park, Hyelin Nam, and Jong Chul Ye. CFG++: Manifold-constrained classifier free guidance for diffusion models. *arXiv preprint arXiv:2406.08070*, 2024.

[8] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[9] Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106 (496):1602–1614, 2011.

[10] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. Scaling Rectified Flow Transformers for High-Resolution Image Synthesis. *arXiv preprint arXiv:2403.03206*, 2024.

[11] C. W. Gear and D. R. Wells. Multirate linear multistep methods. *BIT*, 24(4):484–502, December 1984. ISSN 1572-9125. doi: 10.1007/bf01934907. URL `http://dx.doi.org/10.1007/BF01934907`.

[12] Pareesa Ameneh Golnari, Zhewei Yao, and Yuxiong He. Selective guidance: Are all the denoising steps of guided diffusion important? *arXiv preprint arXiv:2305.09847*, 2023.

[13] Sylvain Gugger, Lysandre Debut, Thomas Wolf, Philipp Schmid, Zachary Mueller, Sourab Mangrulkar, Marc Sun, and Benjamin Bossan. Accelerate: Training and inference at scale made simple, efficient and adaptable. `https://github.com/huggingface/accelerate`, 2022.

[14] Ernst Hairer, Syvert P. Nørsett, and Gerhard Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*, volume 8 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, Heidelberg, 2 edition, 1993. ISBN 978-3-540-56670-0, 978-3-540-78862-1. doi: 10.1007/978-3-540-78862-1.

[15] Moayed Haji-Ali, Guha Balakrishnan, and Vicente Ordonez. Elasticdiffusion: Training-free arbitrary size image generation through global-local content separation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6603–6612, 2024.

[16] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*, 2021.

[17] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[18] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[20] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in Neural Information Processing Systems*, 35:8633–8646, 2022.

[21] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*, 2022.

[22] Sadeep Jayasumana, Srikumar Ramalingam, Andreas Veit, Daniel Glasner, Ayan Chakrabarti, and Sanjiv Kumar. Rethinking fid: Towards a better evaluation metric for image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9307–9315, 2024.

[23] Steffen Jung and Margret Keuper. Internalized biases in fréchet inception distance. In *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2021.

[24] Kevin Kilgour, Mauricio Zuluaga, Dominik Roblek, and Matthew Sharifi. Fréchet audio distance: A reference-free metric for evaluating music enhancement algorithms. In *Proc. Interspeech 2019*, pages 2350–2354, 2019.

[25] Chris Dongjoo Kim, Byeongchang Kim, Hyunmin Lee, and Gunhee Kim. Audiocaps: Generating captions for audios in the wild. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 119–132, 2019.

[26] Tuomas Kynkäänniemi, Miika Aittala, Tero Karras, Samuli Laine, Timo Aila, and Jaakko Lehtinen. Applying guidance in a limited interval improves sample and distribution quality in diffusion models. *arXiv preprint arXiv:2404.07724*, 2024.

[27] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*, pages 740–755. Springer, 2014.

[28] Haohe Liu, Yi Yuan, Xubo Liu, Xinhao Mei, Qiuqiang Kong, Qiao Tian, Yuping Wang, Wenwu Wang, Yuxuan Wang, and Mark D Plumbley. Audioldm 2: Learning holistic audio generation with self-supervised pretraining. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:2871–2883, 2024.

[29] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. *arXiv preprint arXiv:2202.09778*, 2022.

[30] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. 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.

[31] Zhengyao Lv, Chenyang Si, Junhao Song, Zhenyu Yang, Yu Qiao, Ziwei Liu, and Kwan-Yee K Wong. Fastercache: Training-free video diffusion model acceleration with high quality. *arXiv preprint arXiv:2410.19355*, 2024.

[32] Xinyin Ma, Gongfan Fang, Michael Bi Mi, and Xinchao Wang. Learning-to-cache: Accelerating diffusion transformer via layer caching. *Advances in Neural Information Processing Systems*, 37:133282–133304, 2024.

[33] Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15762–15772, 2024.

[34] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14297–14306, 2023.

[35] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On Aliased Resizing and Surprising Subtleties in GAN Evaluation. In *CVPR*, 2022.

[36] A Paszke. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *arXiv preprint arXiv:1912.01703*, 2019.

[37] Sayak Paul. coco-30-val-2014: 30k randomly sampled image-captioned pairs from the COCO 2014 validation split. `https://huggingface.co/datasets/sayakpaul/coco-30-val-2014`, 2023. Dataset on Hugging Face. Accessed 18 Apr 2025.

[38] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[39] John R Rice. Split Runge-Kutta methods for simultaneous equations. *Journal of Research of the National Institute of Standards and Technology*, 60, 1960.

[40] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[41] Ludan Ruan, Yiyang Ma, Huan Yang, Huiguo He, Bei Liu, Jianlong Fu, Nicholas Jing Yuan, Qin Jin, and Baining Guo. Mm-diffusion: Learning multi-modal diffusion models for joint audio and video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10219–10228, 2023.

[42] Seyedmorteza Sadat, Jakob Buhmann, Derek Bradley, Otmar Hilliges, and Romann M Weber. CADS: Unleashing the diversity of diffusion models through condition-annealed sampling. *arXiv preprint arXiv:2310.17347*, 2023.

[43] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

[44] Adrian Sandu. Multirate time integration: an overview. `https://cnls.lanl.gov/tim2023/talks/Sandu_tim2023.pdf`, Aug 2023. Presentation at the 2023 Los Alamos Workshop on Time Integration for Multiphysics. Accessed 30 Apr 2025.

[45] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[46] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[47] Stability AI. Stable Diffusion 3.5 Large. `https://huggingface.co/stabilityai/stable-diffusion-3.5-large`, 2024. Model on Hugging Face. Accessed 25 Apr 2025.

[48] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, William Berman, Yiyi Xu, Steven Liu, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. `https://github.com/huggingface/diffusers`, 2022.

[49] Jingjing Wang, Dan Zhang, and Feng Luo. Unified Directly Denoising for Both Variance Preserving and Variance Exploding Diffusion Models. *arXiv preprint arXiv:2405.21059*, 2024.

[50] Yusong Wu, Ke Chen, Tianyu Zhang, Yuchen Hui, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[51] Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:15903–15935, 2023.

[52] Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all. *arXiv preprint arXiv:2412.20404*, 2024.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims made in the abstract and introduction are stated in Section 3 and Section 4.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitation of our work in Section 5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We specify the assumptions in the preliminaries (Section 3) and the beginning of each algorithm. We also provide complete derivations of our formulas.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide full descriptions of the introduced algorithms.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We publicly release our code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We specify which model, dataset, and hyperparameters are used in the experiments in Section 4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide 2-sigma error ranges for figures in Section 3. Our main table include results from repeated experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide machine specification in Section 4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conform with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss broader impacts of our research in Section 5.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper does not introduce any new models or datasets.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We mention the licenses of the used assets in the appendix.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release any new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A v-prediction models

Recent models such as Stable Diffusion 3.5 [47] directly infer v, or the *velocity field* of the reverse diffusion process. The diffusion ODE is then defined as

$$\frac{\mathrm{d}}{\mathrm{d}t}x_t = \hat{v}_\theta(x_t), \quad x_T \sim \mathcal{N}(0, I). \tag{17}$$

By the definition of CFG [18], we have

$$\hat{v}_\theta(x_t) := \hat{v}_\varnothing(x_t) + \omega \cdot (\hat{v}_c(x_t) - \hat{v}_\varnothing(x_t)) \equiv \hat{v}_c(x_t) + (\omega - 1) \cdot \Delta\hat{v}_c(x_t) \tag{18}$$

where $\Delta\hat{v}_c(x_t) := \hat{v}_c(x_t) - \hat{v}_\varnothing(x_t)$. Substituting Eq. 18 into Eq. 17 yields the following:

$$\frac{\mathrm{d}}{\mathrm{d}t}x_t = \hat{v}_c(x_t) + (\omega - 1) \cdot \Delta\hat{v}_c(x_t). \tag{19}$$

We split this diffusion ODE into a multirate system of ODEs similar to Section 3.1.

$$\frac{\mathrm{d}}{\mathrm{d}t}x_t^{\mathsf{T}} = \hat{v}_c(x_t^{\mathsf{T}} + x_t^{\mathsf{H}}), \quad \frac{\mathrm{d}}{\mathrm{d}t}x_t^{\mathsf{H}} = (\omega - 1) \cdot \Delta\hat{v}_c(x_t^{\mathsf{T}} + x_t^{\mathsf{H}}). \tag{20}$$

Both equations retain the form of Eq. 17 so that existing solvers as the Euler method can be applied to each equation without modification. Furthermore, Algorithm 1 could be utilized unchanged since it is agnostic to the form of equation or the type of the diffusion model solver.

B Proof for approximation error bound analysis

We provide a proof for error accumulation presented in Section 3.3. More rigourous analysis of error bounds could be found in Section II. 3. of [14].

Theorem 1. *Assume the local integration error of an ODE using a solver of order p and timestep size Δt is given by:*

$$\hat{x}_{t-\Delta t} - x_{t-\Delta t} = c \cdot (\Delta t)^{p+1} + \mathcal{O}((\Delta t)^{p+2}) \tag{21}$$

for sufficiently small Δt. Then the error of using the same solver repeatedly for m steps is given by

$$\hat{x}_{t-m\Delta t} - x_{t-m\Delta t} = c \cdot m(\Delta t)^{p+1} + \mathcal{O}((\Delta t)^{p+2}). \tag{22}$$

Proof. We use mathematical induction. (**Base step**) For $m = 1$, Eq. 22 reduces to the assumption. (**Inductive step**) Assume the error of using the solver m times is given by Eq. 22. We proceed to the next iteration to obtain $\hat{x}_{t-(m+1)\Delta t}$. Let $\tilde{x}_{t-(m+1)\Delta t}$ be the *exact* solution given by solving the ODE from $t - m\Delta t$ to $t - (m + 1)\Delta t$ using $\hat{x}_{t-m\Delta t}$. The error in Eq. 22 is transported to the next timestep as

$$\begin{aligned}\tilde{x}_{t-(m+1)\Delta t} - x_{t-(m+1)\Delta t} &= (I + \mathcal{O}(\Delta t))\,(\hat{x}_{t-m\Delta t} - x_{t-m\Delta t}) &(23)\\ &= c \cdot m(\Delta t)^{p+1} + \mathcal{O}((\Delta t)^{p+2}). &(24)\end{aligned}$$

On the other hand, the local error of the next iteration is also given by Eq. 21:

$$\hat{x}_{t-(m+1)\Delta t} - \tilde{x}_{t-(m+1)\Delta t} = c \cdot (\Delta t)^{p+1} + \mathcal{O}((\Delta t)^{p+2}). \tag{25}$$

The error of using the solver $m + 1$ times is thus

$$\hat{x}_{t-(m+1)\Delta t} - x_{t-(m+1)\Delta t} = c \cdot (m + 1)(\Delta t)^{p+1} + \mathcal{O}((\Delta t)^{p+2}). \tag{26}$$

Therefore the error of using the ODE solver m times is given by Eq. 22 for all positive integer m. □

C More details for Richardson Extrapolation

We specify further details about the computation of the coarse timestep grid C. We calculate $\|\hat{x}_s^{\mathsf{T}(1)} - \hat{x}_s^{\mathsf{T}(2)}\|$ and $\|\hat{x}_s^{\mathsf{H}(1)} - \hat{x}_s^{\mathsf{H}(2)}\|$ by solving both the tortoise and hare equations on the fine-grained timestep grid using Algorithm 3. In particular, for each denoising step t_i, we first find $\hat{x}_{t_{i+1}}^{(1)}$ by using the diffusion model solver once from t_i to t_{i+1}. Then we find $\hat{x}_{t_{i+1}}^{(2)}$ by using the diffusion model solver twice, from t_i to $(t_i + t_{i+1})/2$ and from $(t_i + t_{i+1})/2$ to t_{i+1}. We use $\hat{x}_{t_{i+1}}^{(1)}$ for the next denoising step to ensure that we follow the reference trajectory of CFG [18]. Together with Algorithm 2, we obtain the coarse timestep grid C specified in Table 4.

Algorithm 3 Richardson Extrapolation

Require: $x_T \sim \mathcal{N}(0, \sigma_T^2 I)$ ▷ Initial noise
Require: $\omega \geq 0$ ▷ Guidance scale
Require: $\{t_i\}_{0 \leq i \leq N}, t_0 = T, t_N = 0$ ▷ Fine-grained timestep grid

$x_T^{\mathsf{T}} \leftarrow x_T$
$x_T^{\mathsf{H}} \leftarrow 0$
for $i = 0$ **to** $N - 1$ **do**
  $\hat{\epsilon}_c \leftarrow \hat{\epsilon}_\theta(x_{t_i}^{\mathsf{T}} + x_{t_i}^{\mathsf{H}}, c)$
  $\hat{\epsilon}_\varnothing \leftarrow \hat{\epsilon}_\theta(x_{t_i}^{\mathsf{T}} + x_{t_i}^{\mathsf{H}}, \varnothing)$
  $\Delta\hat{\epsilon}_c \leftarrow \hat{\epsilon}_c - \hat{\epsilon}_\varnothing$
  $\hat{x}_{t_{i+1}}^{\mathsf{T}(1)} \leftarrow \text{Solver}(x_{t_i}^{\mathsf{T}}, \hat{\epsilon}_c, t_i, t_{i+1})$ ▷ $\hat{x}_{t_{i+1}}^{(1)}$ of the tortoise
  $\hat{x}_{t_{i+1}}^{\mathsf{H}(1)} \leftarrow \text{Solver}(x_{t_i}^{\mathsf{H}}, (\omega - 1) \cdot \Delta\hat{\epsilon}_c, t_i, t_{i+1})$ ▷ $\hat{x}_{t_{i+1}}^{(1)}$ of the hare
  $t_m = (t_i + t_{i+1})/2$ ▷ Midpoint of current and next timesteps
  $\hat{x}_{t_m}^{\mathsf{T}(2)} \leftarrow \text{Solver}(x_{t_i}^{\mathsf{T}}, \hat{\epsilon}_c, t_i, t_m)$
  $\hat{x}_{t_m}^{\mathsf{H}(2)} \leftarrow \text{Solver}(x_{t_i}^{\mathsf{H}}, (\omega - 1) \cdot \Delta\hat{\epsilon}_c, t_i, t_m)$
  $\hat{\epsilon}_c \leftarrow \hat{\epsilon}_\theta\left(\hat{x}_{t_m}^{\mathsf{T}(2)} + \hat{x}_{t_m}^{\mathsf{H}(2)}, c\right)$
  $\hat{\epsilon}_\varnothing \leftarrow \hat{\epsilon}_\theta\left(\hat{x}_{t_m}^{\mathsf{T}(2)} + \hat{x}_{t_m}^{\mathsf{H}(2)}, \varnothing\right)$
  $\Delta\hat{\epsilon}_c \leftarrow \hat{\epsilon}_c - \hat{\epsilon}_\varnothing$
  $\hat{x}_{t_{i+1}}^{\mathsf{T}(2)} \leftarrow \text{Solver}(\hat{x}_{t_m}^{\mathsf{T}(2)}, \hat{\epsilon}_c, t_m, t_{i+1})$ ▷ $\hat{x}_{t_{i+1}}^{(2)}$ of the tortoise
  $\hat{x}_{t_{i+1}}^{\mathsf{H}(2)} \leftarrow \text{Solver}(\hat{x}_{t_m}^{\mathsf{H}(2)}, (\omega - 1) \cdot \Delta\hat{\epsilon}_c, t_m, t_{i+1})$ ▷ $\hat{x}_{t_{i+1}}^{(2)}$ of the hare
  $x_{t_{i+1}}^{\mathsf{T}} \leftarrow \hat{x}_{t_{i+1}}^{\mathsf{T}(1)}$ ▷ Tortoise of next step
  $x_{t_{i+1}}^{\mathsf{H}} \leftarrow \hat{x}_{t_{i+1}}^{\mathsf{H}(1)}$ ▷ Hare of next step
end for
return $\|\hat{x}_{t_{i+1}}^{\mathsf{T}(1)} - \hat{x}_{t_{i+1}}^{\mathsf{T}(2)}\|, \|\hat{x}_{t_{i+1}}^{\mathsf{H}(1)} - \hat{x}_{t_{i+1}}^{\mathsf{H}(2)}\|$

Table 4: Obtained coarse timestep grid for different ρ values. Our choice is marked in blue. For brevity, only indices of the timesteps are shown. Note that only $i < i_{\text{hi}}$ is actually used in the final algorithm.

ρ	$\{i \mid t_i \in C\}$
Stable Diffusion 1.5 with DDIM	
0.9	{0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49}
1.0	{0, 1, 2, 3, 4, 5, 6, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49}
1.1	{0, 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 17, 20, 23, 26, 28, 30, 32, 34, 36, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49}
1.2	{0, 1, 2, 3, 4, 5, 7, 9, 11, 14, 17, 20, 23, 26, 29, 31, 33, 35, 37, 39, 41, 42, 43, 44, 45, 46, 47, 48, 49}
1.3	{0, 1, 2, 3, 4, 6, 8, 10, 13, 16, 19, 22, 25, 28, 31, 34, 36, 38, 40, 42, 44, 45, 46, 47, 48, 49}
Stable Diffusion 3.5 Large with Euler method	
0.9	{0, 1, 2, 3, 5, 7, 10, 13, 16, 18, 20, 22, 23, 24, 25, 26}
1.0	{0, 1, 2, 4, 6, 9, 12, 15, 18, 20, 22, 23, 24, 25, 26}
1.1	{0, 1, 2, 4, 6, 9, 13, 17, 20, 22, 23, 24, 25, 27}
1.2	{0, 1, 2, 4, 7, 11, 15, 19, 21, 23, 25, 27}
1.3	{0, 1, 3, 6, 10, 15, 19, 22, 24, 26}

Table 5: Obtained coarse timestep grid for different ω values. Our choice is marked in blue. For brevity, only indices of the timesteps are shown. Note that only $i < i_{\text{hi}}$ is actually used in the final algorithm. The results show that while a bigger ω results in a denser C, the overall trend is consistent.

Variant	$\{i \mid t_i \in C\}$
Stable Diffusion 1.5 with DDIM	
$\omega = 6.5, \rho = 0.93$	{0, 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 17, 20, 23, 26, 28, 30, 32, 34, 36, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50}
$\omega = 6.5, \rho = 1.1$	{0, 1, 2, 3, 4, 6, 8, 10, 13, 16, 19, 22, 25, 28, 31, 33, 35, 37, 39, 41, 43, 44, 45, 46, 47, 48, 49, 50}
$\omega = 7.5, \rho = 1.1$	{0, 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 17, 20, 23, 26, 28, 30, 32, 34, 36, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50}
$\omega = 8.5, \rho = 1.1$	{0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50}
$\omega = 8.5, \rho = 1.22$	{0, 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 17, 20, 23, 26, 28, 30, 32, 34, 36, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50}

Table 6: Obtained coarse timestep grid with fewer sample trajectories. Our original choice is marked in blue. For brevity, only indices of the timesteps are shown. Note that only $i < i_{\text{hi}}$ is actually used in the final algorithm. The results show that C can be computed accurately with only 1,000 sample trajectories.

Batch size	IoU	$\{i \mid t_i \in C\}$
Stable Diffusion 1.5 with DDIM		
30k	–	{0, 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 17, 20, 23, 26, 28, 30, 32, 34, 36, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50}
1k (trial 1)	100%	{0, 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 17, 20, 23, 26, 28, 30, 32, 34, 36, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50}
1k (trial 2)	100%	{0, 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 17, 20, 23, 26, 28, 30, 32, 34, 36, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50}
1k (trial 3)	100%	{0, 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 17, 20, 23, 26, 28, 30, 32, 34, 36, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50}
		...
1k (trial 8)	57.5%	{0, 1, 2, 3, 4, 5, 7, 9, 11, 13, 16, 19, 22, 25, 28, 30, 32, 34, 36, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50}
		...
1k (trial 30)	100%	{0, 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 17, 20, 23, 26, 28, 30, 32, 34, 36, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50}

D Sensitivity of C

Guidance weight ω The coarse timestep grid C depends on guidance weight ω, since the error bounds computed by Algorithm 3 depend on ω. As ω increases, both the hare term and its approximation error grow, resulting in a denser coarse grid C. Table 5 shows C evaluated under different ω values. While a bigger guidance scale results in a denser C, the overall trend is consistent; one can obtain the same C by adjusting ρ.

Batch size While we computed C with sample trajectories on 30,000 prompts from the COCO 2014 dataset, it is possible to compute C using fewer sample trajectories. Table 6 shows C computed on a batch of 1,000 prompts, compared to C computed on 30,000 prompts. The results closely matched our original, large-scale estimate. Compared to the original estimate, the 1,000 sample estimate demonstrated 95.25% IoU (i.e., Jaccard index) in average over 30 trials.

Table 7: Comparison of methods in terms of distributional similarity and prompt fidelity. Our method is marked in blue, whereas vanilla CFG is marked in gray. The parameters of THG correspond to (ρ, b, i_{hi}). The best results are **highlighted**.

Method	N	NFE ↓	Distributional similarity FID ↓	CMMD ↓	Prompt fidelity CS ↑	IR ↑
Stable Diffusion 1.5 with DDIM						
CFG [18]	50	100	14.133	0.58948	26.295	0.14764
Selective Guidance [12]	50	70	**12.895**	**0.56052**	25.602	−0.06691
Guidance Interval [26]	50	70	14.555	0.62227	26.153	0.09658
CFG-Cache [31]	50	70	14.422	0.59759	26.179	0.09705
THG (1.1, 1.1, 38)	50	70	14.232	0.59354	**26.197**	**0.11576**
CFG [18]	35	70	13.342	0.57232	26.321	0.12694
Selective Guidance [12]	35	49	**12.299**	**0.54550**	25.623	−0.09007
Guidance Interval [26]	35	49	14.490	0.61178	26.140	0.05919
CFG-Cache [31]	35	49	13.773	0.58162	26.120	0.05246
THG (1.7, 1.1, 30)	35	49	13.468	0.57637	**26.265**	**0.09202**
CFG [18]	20	40	13.366	0.56159	26.370	0.10090
Selective Guidance [12]	20	30	**12.839**	**0.54740**	25.900	−0.04452
Guidance Interval [26]	20	30	14.937	0.61333	26.244	0.02827
CFG-Cache [31]	20	30	13.675	0.56815	26.211	0.04604
THG (3.5, 1.1, 17)	20	30	13.345	0.56719	**26.278**	**0.07212**

E More comparisons with CFG variants

Table 7 extends our comparison to include additonal baselines [12, 26] across a wider range of NFEs. Note that we used a different number of fine-grained timesteps N to control the NFE of CFG. The results demonstrate that our approach achieves superior image quality with the same NFE budget. While Selective Guidance [12] shows lower FID and CMMD values, they generate images with low prompt fidelity and degraded details indicated by lower CS and IR values as noted in Appendix G. Aside from this phenomenon, THG achieves superior overall performance.

Comparison with vanilla CFG Although vanilla CFG with 70 NFE yields a FID of 13.342 (better than THG's 14.232 with 70 NFE), its PSNR drops substantially to 19.42 dB compared to 24.16 dB for THG. This aligns with observations in [6], where reducing N sometimes lowers FID. Moreover, [35, 23] show that FID may not reliably reflect perceptual quality when the image structures diverge, so we focus on comparisons at the same N.

F More details for metrics

FID, FAD Fréchet Inception Distance [17] (FID) is a ubiquitously used metric for developing and adopting image generative models. It measures the distance between real and generated images in a deep feature space to capture relevant features of the two distributions [35]. Therefore, a lower FID value indicates realistic generation. To measure FID, one first uses an InceptionV3 feature extractor model to compute features from real and generated images. Under the assumption that the resulting feature sets follows a multidimensional Gaussian distribution, the distance of the two distributions $\mathcal{N}(\mu_r, \Sigma_r)$ and $\mathcal{N}(\mu_g, \Sigma_g)$ is given by the Fréchet distance

$$\text{FD} = ||\mu_r - \mu_g||_2^2 + \text{tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}). \tag{27}$$

Similarly, Fréchet Audio Distance [24] (FAD) measures the distance between real and generated audio by using a VGGish embedding model to extract features from audio clips.

CMMD CMMD [22] is a recently proposed alternative for FID. It uses CLIP [38] embeddings and the Maximum Mean Discrepancy (MMD) distance instead of InceptionV3 features and the Fréchet distance. CLIP is trained on 400 million images with corresponding text descriptions and therefore

Table 8: Ablation study for the guidance scale ω with CFG [18].

Method	NFE ↓	FID ↓	CMMD ↓	CS ↑	IR ↑
Stable Diffusion 1.5 with DDIM					
$\omega = 2.5$	100	8.438	0.56672	25.153	-0.28577
$\omega = 3.5$	100	9.143	0.54192	25.687	-0.09190
$\omega = 4.5$	100	10.644	0.54764	25.935	0.00670
$\omega = 5.5$	100	12.030	0.56171	26.110	0.07195
$\omega = 6.5$	100	13.222	0.57673	26.225	0.11582
$\omega = 7.5$	100	14.133	0.58948	26.295	0.14764
$\omega = 8.5$	100	14.902	0.60343	26.369	0.17431

Figure 5: **Generated images using** $\omega = 2.5$ for the prompts "A group of zebras grazing in the grass.", "A yellow commuter train traveling past some houses.", "A couple of men standing on a field playing baseball.", and "Zoo scene of children at zoo near giraffes, attempting to pet or feed them." from the COCO 2014 dataset.

much more suitable for capturing rich and diverse content. MMD does not make any assumptions about the underlying distributions, unlike the Fréchet distance which assumes multidimensional Gaussian distributions. By combining CLIP and MMD, CMMD avoids the drawbacks of FID.

CLIP Score, CLAP Score CLIP Score [16] uses CLIP [38] to assess image-caption compatibility. CLIP learns a multimodal embedding space by jointly traning an image encoder and a text encoder, while the cosine similarity of matching image and text embeddings are maximized. Leveraging this design, CLIP score is defined by the cosine similarity of the image embedding E_I and text prompt embedding E_C as

$$\text{CLIPScore}(\text{I}, \text{C}) = \max(100 \cdot \cos(E_I, E_C), 0). \tag{28}$$

We report the average CLIP score over the generated image set. Similarly, CLAP Score [50] uses the CLAP model to assess audio-caption compatibility.

ImageReward ImageReward [51] is a general-purpose text-to-image human preference reward model, effectively encoding human preferences by traning on actual human feedback. Experiments shows that ImageReward aligns with human ranking better than zero-shot FID and CLIP Score. Also, ImageReward values have larger quantile value range than that of CLIP Score, demonstrating that it can well distinguish the quality of images from each other. We report the average ImageReward value over the generated image set.

G Tradeoff of distributional similarity and prompt fidelity

Tables 1 and 2 demonstrate a tradeoff between distributional similarity metrics (FID, CMMD) and prompt fidelity metrics (CS, IR). When the prompt fidelity metrics improve so that each image matches better with the given prompt, the distributional similarity metrics worsen so that the distribution of the images is further from that of real images.

We further investigate this phenomenon by conducting an additional ablation study for the guidance scale ω using Stable Diffusion 1.5 and CFG. Table 8 shows how the metrics change as ω is changed. The minimum FID is achieved at $\omega = 2.5$ and the minimum CMMD is achieved at $\omega = 3.5$. However,

they suffer from low CS and IR. Generated images using $\omega = 2.5$ are visualized in Fig. 5, showing degraded details or insufficient text alignment. This suggests that lower FID or CMMD does not always indicate better generation quality. While these distributional similarity metrics measure both image plausibility and diversity, they can possibly fail to report high-quality details of the images with lower values.

Since the global structure of each image is determined by the initial few steps of the reverse diffusion process [26, 3], the images generated by the methods in Table 1 have mostly shared global structures and differ on delicate details. Given that, we suggest that the human-perceived quality of generated samples could be better explained by the prompt fidelity metrics compared to the distributional similarity metrics. Our results in Table 1 with slightly higher FID or CMMD therefore do not indicate a significant degradation of generation quality.

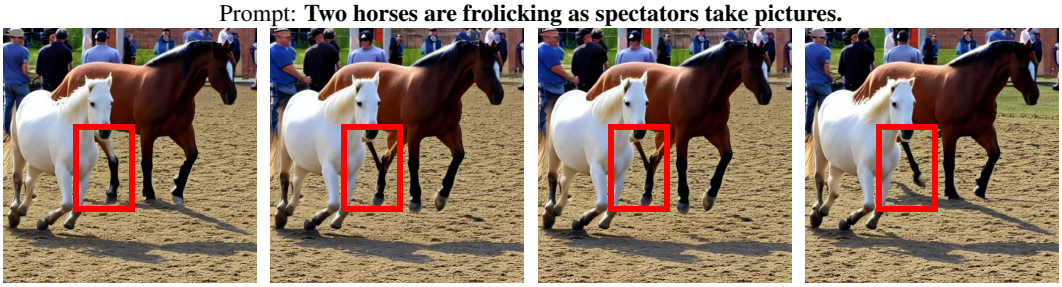
H Licenses

- **Stable Diffusion 1.5** – weights released under the CreativeML Open RAIL-M license (v1.0; `https://github.com/CompVis/stable-diffusion/blob/main/LICENSE`)
- **Stable Diffusion 3.5 Large** – weights released under the Stability AI Community Licence v3 (research & commercial use for organizations or individuals with < USD 1 M annual revenue; `https://stability.ai/license`)
- **FID** – clean-FID implementation by Parmar et al., released under the MIT License (v1.0; `https://github.com/GaParmar/clean-fid/blob/main/LICENSE`)
- **CMMD** – PyTorch implementation of CLIP Maximum Mean Discrepancy by Sayak Paul, released under the Apache License 2.0 (v2.0; `https://github.com/sayakpaul/cmmd-pytorch/blob/main/LICENSE`)
- **CLIP Score** – TorchMetrics' CLIPScore module released under the Apache License 2.0 (v2.0; `https://github.com/Lightning-AI/metrics/blob/master/LICENSE`; Lightning-AI)
- **ImageReward** – model and evaluation code released under the Apache License 2.0 (v2.0; `https://github.com/THUDM/ImageReward/blob/main/LICENSE`; Xu et al., 2023)
- **MS COCO 2014**:
  - Annotations released under the Creative Commons Attribution 4.0 International license (CC BY 4.0; `https://creativecommons.org/licenses/by/4.0/`)
  - Underlying images governed by Flickr Terms of Use; users must comply with Flickr's rules when reusing or redistributing any COCO images.
- **AudioLDM 2** – weights released under the Creative Commons Attribution–NonCommercial–ShareAlike 4.0 International License (`https://github.com/haoheliu/AudioLDM2/blob/main/LICENSE`)
- **AudioCaps** – dataset released under the MIT License (v1.0; `https://github.com/cdjkim/audiocaps/blob/master/LICENSE`)
- **FAD** – PyTorch implementation of Frechet Audio Distance by Hao Hao Tan, released under the MIT License (v1.0; `https://github.com/gudgud96/frechet-audio-distance/blob/main/LICENSE`)
- **CLAP Score** – model and evaluation code released under the Creative Commons CC0 1.0 Universal License; public domain dedication (`https://github.com/LAION-AI/CLAP/blob/main/LICENSE`)

I More qualitative results

Figure 6 shows more qualitative results for Stable Diffusion 1.5. Figures 7 and 8 show more qualitative results for Stable Diffusion 3.5 Large.

Prompt: **Two horses are frolicking as spectators take pictures.**



Prompt: **a male with a purple jacket on skies posing for a picture**

(a) CFG (baseline) NFE = 100 | (b) CFG-Cache w/o FFT NFE = 70 | (c) CFG-Cache NFE = 70 | (d) Ours NFE = 70

Figure 6: **Comparison of visual results** for prompts from the COCO 2014 dataset using Stable Diffusion 1.5.

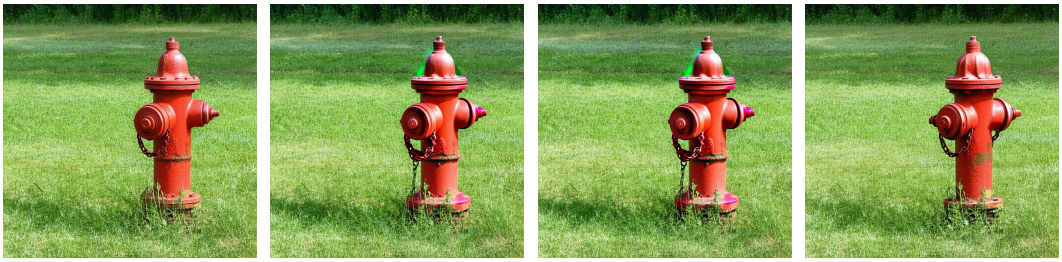
Prompt: **A woman and a man are playing the nintendo wii video game system**

Prompt: **A cat sitting on a window sill near a basket.**

(a) CFG (baseline) NFE = 56 | (b) CFG-Cache w/o FFT NFE = 38 | (c) CFG-Cache NFE = 38 | (d) Ours NFE = 38

Figure 7: **Comparison of visual results** for prompts from the COCO 2014 dataset using Stable Diffusion 3.5 Large.

Prompt: **A red fire hydrant is set up in a grassy clearing.**



Prompt: **A stop sign and one way sign are in front of a large building**

(a) CFG (baseline) NFE = 56 | (b) CFG-Cache w/o FFT NFE = 38 | (c) CFG-Cache NFE = 38 | (d) Ours NFE = 38

Figure 8: **Comparison of visual results** for prompts from the COCO 2014 dataset using Stable Diffusion 3.5 Large.