SIMPLE ENDPOINT REWARD FINETUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reward fine-tuning adapts pretrained diffusion and flow models to desired objectives by optimizing a specified reward. For differentiable rewards, transporting reward gradients backwards along trajectories enables gradient computation at noisy states. However, implementing this in practice is prohibitively expensive; existing first order methods imitating this must make aggressive approximations, often estimating the trajectory through few-step samples and evaluating reward gradients off-policy. This motivates the development of a method using only reward gradients at clean samples, while still approximating the true gradient direction. To this end, we introduce SIMPLE ENDPOINT REWARD FINETUNING (SERF), a remarkably simple fine-tuning objective that avoids gradient transport by directly using the reward gradient at the endpoint. Specifically, at renoised states the trained velocity is regressed against the reference velocity shifted by the clean image's reward gradient. Although easy to state and implement, we show that SERF has a rigorous derivation through stochastic optimal control, and that our objective provides a consistent estimate of the true reward gradient direction. When compared with existing methods, SERF achieves the most favorable trade-offs when used to post-train Stable Diffusion 3.5 on PickScore and HPSv2.1, demonstrating its ability to learn rewards while maintaining diverse outputs. SERF is shown to be highly efficient, up to $10\times$ faster than Adjoint Matching, while remaining flexible enough to allow endpoint sampling in as few as 5 steps.

1 INTRODUCTION

Diffusion and flow-matching models (Ho et al., 2020b; Song et al., 2021; Lipman et al., 2023; Esser et al., 2024; Peluchetti, 2021; Albergo et al., 2025; Liu et al., 2022) have become a dominant paradigm for generative modeling in continuous domains, powering modern text-to-image generation (T2I) (Rombach et al., 2022; Esser et al., 2024), video generation (Ho et al., 2022; Blattmann et al., 2023), and other multimodal applications (Liu et al., 2023; Bao et al., 2023). Such models are trained to reproduce the data distribution by learning a drift that transports a a simple source distribution to the data distribution (Song et al., 2021; Lipman et al., 2023). Once trained, sampling is performed by simulating the learned transport dynamics.

The pretraining of large generative models, including language and flow models, is typically centered around accurately modeling the data distribution, rather than optimizing for downstream performance (Brown et al., 2020). Deployed models are therefore often *reward-finetuned* for specific tasks such as prompt adherence, or human preferences (Ouyang et al., 2022; Xu et al., 2023) For flow models, rewards are usually defined on clean samples, yet fine-tuning must improve the dynamics at noisy intermediate states. As these models generate samples through differentiable dynamics, *differentiable rewards* may be backpropagated from a clean sample back along its trajectory, providing gradient signal at noisy states. However, generating a sample with a flow model typically requires a large number of neural function evaluations (NFEs) of the learned drift (Tong et al., 2024; Geng et al., 2025; Frans et al., 2025), and so naive backpropagation through time (BTT) along the entire sampling trajectory can create a prohibitively large computational graph and can yield unstable gradients (Bergmeister et al., 2026; Clark et al., 2024; Prabhudesai et al., 2023).

Several methods seek to propagate reward information from clean samples to noisy states without the full cost of BTT. Some approaches reduce this cost by stopping gradients through earlier sampling steps(Clark et al., 2024; Prabhudesai et al., 2023), while others use adjoint techniques derived from stochastic optimal control (SOC) (Domingo-Enrich et al., 2025; Shin et al., 2026). Additional

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

approaches learn auxiliary networks to estimate intermediate guidance directions (Liu et al., 2025a; Jia et al., 2025), or form cheap but noisy estimates of the gradient direction by taking reward gradients at few-step roll-outs(Xu et al., 2023; Kang et al., 2026).

Since methods that try to compute the exact reward gradient appear to face considerable computational difficulty, while cheaper alternatives use noisy reward gradients at predicted rather than fully denoised samples that can lead to reward hacking, a natural question is then the following:

> Can we formulate a cheap reward fine-tuning objective that uses only reward gradients at clean samples, while still approximating the true gradient direction at noisy samples?

To answer this question, we introduce SIMPLE ENDPOINT REWARD FINETUNING (SERF), a first-order reward fine-tuning method based on a simple regression objective. Rather than explicitly calculating the update direction at a given noisy state, we first generate clean images with an ODE sampler and use their reward gradients to define a reward direction. We then noise each image to obtain intermediate states and regress the model's velocity toward the reference velocity plus this reward direction (see Figure 1).

By rearranging terms in the usual SOC problem, we show how SERF arises naturally as an approximation of the reward-ascent direction. It is derived from the simple assumption that the change in the endpoint caused by a perturbation of the noised state is the identity. In comparison to other methods which simulate the forwards process during fine-tuning, SERF has provably lower gradient variance.

SERF enjoys many advantages compared to alternative fine-tuning methods. It computes reward gradients on-policy, and is agnostic to the method used for endpoint sampling, since we do not require storage of, nor backpropagation along, the sampling trajectory. Each endpoint reward gradient can supervise many independently re-noised states, decoupling the number of training states from the number of steps used to generate the endpoint.

We evaluate SERF by fine-tuning Stable Diffusion 3.5 (Esser et al., 2024) on HPSv2.1 (Wu et al., 2023) and PickScore (Kirstain et al., 2023) and compare its performance against a comprehensive set of baselines. We find that it achieves better trade-offs between optimized reward, held-out reward scores, image fidelity, and diversity, while remaining effective when generated images are reused for multiple training targets and images are sampled with only a few ODE steps. Additionally, SERF is very efficient, enabling the fine-tuning of Stable Diffusion 3.5 $10\times$ faster than Adjoint Matching (Domingo-Enrich et al., 2025).

2 BACKGROUND

2.1 FLOW AND DIFFUSION MODELS

Let $X_0 \sim p_0 = \mathcal{N}(0, I)$ and $X_1 \sim p_1 = p_{\text{data}}$ be independent, and define the stochastic interpolant as $X_t = (1-t)X_0 + tX_1$. For this linear path, flow matching learns a reference velocity by minimizing

$$\mathcal{L}_{\text{FM}}(v) = \mathbb{E}_{t\sim U[0,1],\, X_0\sim p_0, X_1\sim p_1} \left\| v_t(X_t) - (X_1 - X_0) \right\|^2. \tag{1}$$

Its population minimizer is $v_t^{\text{ref}}(x) = \mathbb{E}[X_1 - X_0 \mid X_t = x]$ (Lipman et al., 2023; Esser et al., 2024). At this optimum, the ODE $\mathrm{d}x_t = v_t^{\text{ref}}(x_t)\mathrm{d}t$, initialized with $x_0 \sim p_0$, has time marginals $p_t := \text{Law}(X_t)$. In particular, its terminal state x_1 is distributed according to p_{data}.

The same marginal path p_t can also be generated by an SDE (Albergo et al., 2025). For the linear interpolation above, one such process is

$$\mathrm{d}X_t^{\text{ref}} = b_t^{\text{ref}}(X_t^{\text{ref}})\mathrm{d}t + \sigma_t \mathrm{d}W_t, \qquad b_t^{\text{ref}}(x) = 2v_t^{\text{ref}}(x) - \frac{x}{t}, \qquad \sigma_t^2 = \frac{2(1-t)}{t}, \tag{2}$$

for $0 < t < 1$ (Song et al., 2021). This choice is memoryless: $X_0^{\text{ref}} \perp X_1^{\text{ref}}$ (Domingo-Enrich et al., 2025). Moreover, its conditional law given the endpoint is the Gaussian noising kernel of the training interpolant,

$$\text{Law}(X_t^{\text{ref}} \mid X_1^{\text{ref}} = x_1) = \text{Law}(X_t \mid X_1 = x_1) = \mathcal{N}(tx_1, (1-t)^2 I).$$

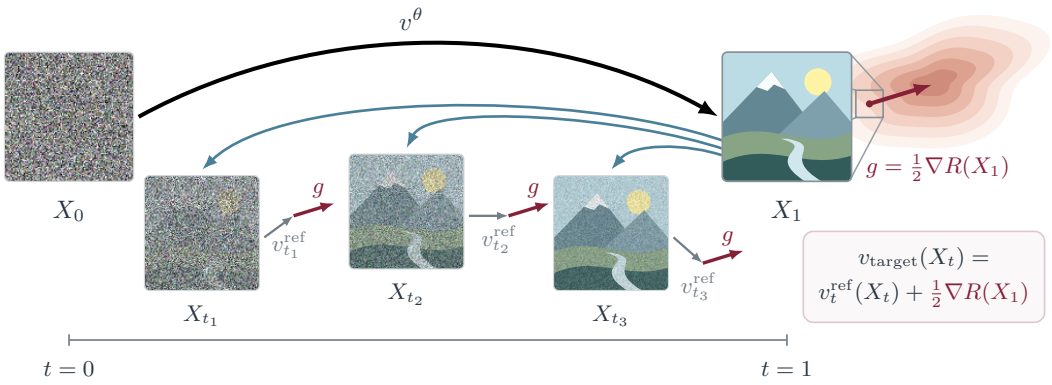

Figure 1: Overview of SERF, which fine-tunes diffusion and flow models by regressing the model's velocity to the sum of the reference velocity and the reward gradient of the generated data.

2.2 Reward fine-tuning

Following Domingo-Enrich et al. (2025), we formulate reward fine-tuning as a stochastic optimal control problem. Let the reference diffusion satisfy $\mathrm{d}X_t^{\text{ref}} = b_t^{\text{ref}}(X_t^{\text{ref}})\,\mathrm{d}t + \sigma_t\,\mathrm{d}W_t$, with $X_0^{\text{ref}} \sim p_0$ and terminal distribution $p_1^{\text{ref}} = p_{\text{data}}$. We seek a controlled drift b, while keeping p_0 and σ_t fixed, that maximizes the stochastic optimal control (SOC) problem

$$\mathcal{L}(b) := \mathbb{E}\left[R(X_1^b) - \frac{1}{2}\int_0^1 w(t)\|b_t(X_t^b) - b_t^{\text{ref}}(X_t^b)\|^2 \mathrm{d}t\right], \tag{3}$$

where the controlled process satisfies $\mathrm{d}X_t^b = b_t(X_t^b)\,\mathrm{d}t + \sigma_t\,\mathrm{d}W_t$ with $X_0^b \sim p_0$. Here, R is the terminal reward and $w(t)$ controls the strength of the reference anchor.

For the choice $w(t) = \sigma_t^{-2}$, the running cost equals the Kullback–Leibler divergence between the controlled and reference path measures (Berner et al., 2022). In this setting, the optimal drift $b_t^\star$ induces the endpoint-tilted path measure, with terminal marginal $p_1^{b^\star}(x) \propto p_{\text{data}} e^{R(x)}$ (Domingo-Enrich et al., 2025). Since the tilt depends only on the endpoint, the conditional path laws coincide:

$$\text{Law}\left((X_t^{b^\star})_{t\in[0,1]} \,\middle|\, X_1^{b^\star} = x\right) = \text{Law}\left((X_t^{\text{ref}})_{t\in[0,1]} \,\middle|\, X_1^{\text{ref}} = x\right). \tag{4}$$

3 Efficient first-order training

Here we aim to develop Simple Endpoint Reward Finetuning to approximately solve the SOC problem Equation (3). We motivate it by deriving a surrogate target leveraging the structure of the known optimal solution of Equation (4), which is approximated under assumptions on the stochastic Jacobian of the generative SDE.

SERF first generates a clean sample and computes its reward gradient, then uses this gradient together with the reference velocity as an approximate guidance term for intermediate re-noised data points, which leads to the following loss:

$$\begin{aligned} \mathcal{L}_{\text{SERF}}(v) &= \mathbb{E}_{t\sim U[0,1]} \left\| v_t(X_t) - v_t^{\text{ref}}(X_t) - \tfrac{1}{2}\nabla R(X_1) \right\|^2, \\ X_1 \sim p_1^v &:= (\Phi_{0,1}^v)_{\#} p_0, \qquad X_t \mid X_1 \sim \mathcal{N}\big(tX_1, (1-t)^2 I\big). \end{aligned} \tag{5}$$

The overall algorithm is summarized in Algorithm 1. Figure 1 details a single SERF training step: the reward gradient tilts the on-policy velocity while the reference velocity serves as an anchor. Although we formulate SERF in terms of fine-tuning the drift b under SOC to ease the exposion, in practice our algorithm only trains the deterministic velocity v.

Loss decomposition. When b is parameterized by a neural network, directly optimizing the SOC objective $\mathcal{L}(b)$ involves costly backpropagation through the sampled SDE trajectories. Instead, we

Algorithm 1 Simple Endpoint Reward Finetuning (SERF)

Require: Pretrained flow model v^{ref}, differentiable reward R, outer steps K, inner steps L
Initialize $v^\theta = v^{\text{ref}}$
for $k = 0, \dots, K-1$ **do**
  Sample endpoint minibatch $\{X_1^i\}$ using v^θ
  Compute and store reward gradients $R^i = \nabla R(X_1^i)$
  for $l = 0, \dots, L-1$ **do**
    Sample intermediate states $t^i \sim U[0,1]$, $X^i \sim \mathcal{N}(t^i X_1^i, (1-t^i)^2 I)$
    Compute loss $\mathcal{L}(\theta) = \sum_i \|v_{t^i}^\theta(X^i) - v_{t^i}^{\text{ref}}(X^i) - R^i\|^2$
    Update θ using $\nabla_\theta \mathcal{L}(\theta)$
  end for
end for
return v^θ

first decompose the objective into a reward term, $\mathcal{L}_{\text{reward}}$, and a regularization one $\mathcal{L}_{\text{regularization}}$, with the aim of deriving an efficient surrogate for each:

$$\mathcal{L}(b) = \underbrace{\mathbb{E}[R(X_1^b)]}_{\mathcal{L}_{\text{reward}}(b)} - \underbrace{\frac{1}{2}\mathbb{E}\left[\int_0^1 w(t)\left\|b_t(X_t^b) - b_t^{\text{ref}}(X_t^b)\right\|^2 \mathrm{d}t\right]}_{\mathcal{L}_{\text{regularization}}(b)}. \tag{6}$$

Reward ascent. Maximizing $\mathcal{L}_{\text{reward}}$ has an equivalent formulation as an L_2 loss:

$$\hat{\mathcal{L}}_{\text{reward}}(b) = \frac{1}{2}\mathbb{E}\left[\int_0^1 \left\|b_t(X_t^{\bar{b}}) - \bar{b}_t(X_t^{\bar{b}}) - \nabla_{X_t^{\bar{b}}}\mathbb{E}[R(X_1^{\bar{b}}) \mid X_t^{\bar{b}}]\right\|^2 \mathrm{d}t\right], \tag{7}$$

where $\bar{b} = \text{stopgradient}(b)$ denotes that the derivative of $\bar{b}$ with respect to θ is set to 0.

Theorem 1. *The gradient of $\mathcal{L}_{\text{reward}}(b)$ with respect to the parameters θ of the drift b is equal to the negative gradient of $\hat{\mathcal{L}}_{\text{reward}}(b)$.*

We prove this in §A.1. This shows maximizing $\mathcal{L}_{\text{reward}}$ is equivalent to regressing b_t in the direction of the gradient of the expected reward. Unfortunately, standard unbiased estimates for the regression target require simulation of the entire diffusion trajectory (Domingo-Enrich et al., 2025).

Efficient first-order estimation. We approximate the intractable target in Equation (7) as

$$\nabla_x \mathbb{E}\left[R(X_1^{\bar{b}}) \mid X_t^{\bar{b}} = x\right] = \mathbb{E}\left[\left(J_{1,t}^{\bar{b}}\right)^\top \nabla R(X_1^{\bar{b}}) \mid X_t^{\bar{b}} = x\right] \approx \mathbb{E}\left[\nabla R(X_1^{\bar{b}}) \mid X_t^{\bar{b}} = x\right] \tag{8}$$

where $J_{1,t}^{\bar{b}}$ is the stochastic Jacobian, which describes how perturbations of $X_t^{\bar{b}}$ propagate to $X_1^{\bar{b}}$, see §A.2. We approximate it by the identity matrix, as successfully done for reward steering (McAllister et al., 2026) and fine-tuning with non-differentiable rewards (Zhou et al., 2026). Under the conditions stated below, the resulting surrogate gradient remains positively aligned with the target in Equation (7):

Proposition 1. *$\hat{R}_t(x) := \mathbb{E}[\nabla R(X_1^{\bar{b}}) \mid X_t^{\bar{b}} = x]$ is positively aligned with $\nabla_x \mathbb{E}[R(X_1) \mid X_t = x]$ if $\text{Law}(X_1 \mid X_t = x) = \mathcal{N}(T(x), \sigma^2 I)$, where $J_T(x) = \nabla_x T(x)$ has positive-definite symmetric part. In particular, if $\text{Law}(X_t \mid X_1 = x)$ is Gaussian, then this holds for $T(x) = \mathbb{E}[X_1 \mid X_t = x]$.*

Thus, Proposition 1 holds when the diffusion model's conditional distribution is approximated by a Gaussian. Using $\hat{R}$ as an efficient proxy for the expected reward gradient in $\hat{\mathcal{L}}_{\text{reward}}$ yields the following simplified reward objective (see §A.3 for the derivation):

$$\mathcal{L}_{\text{simple-reward}}(b) = \frac{1}{2}\mathbb{E}_{t\sim U[0,1]}\left\|b_t(X_t^{\bar{b}}) - \bar{b}_t(X_t^{\bar{b}}) - \nabla R(X_1^{\bar{b}})\right\|^2, \tag{9}$$

which in particular, it is still a reward ascent direction (§A.3).

Regularization. The regularization term $\mathcal{L}_{\text{regularization}}$ in Equation (6) regresses b_t towards the reference b_t^{ref} at states X_t^b, with gradients propagated through the sampling process. For computational efficiency, we instead sample X_t using $\text{stopgradient}(b)$, treating the sampling distribution as fixed during differentiation, as is common in fine-tuning methods (Xue et al., 2026; Choi et al., 2026a).

Choosing the loss weighting $w(t) = \sigma(t)^{-2}$ gives $\mathcal{L}_{\text{regularization}}$ the interpretation of a path KL divergence (Berner et al., 2022). Although this weighting is theoretically motivated, constant or otherwise modified time weightings often give better empirical performance (Ho et al., 2020a; Zheng et al., 2026a; Choi et al., 2026a). We therefore use $w(t) = 1$, which yields the simplified regularization loss

$$\mathcal{L}_{\text{simple-regularization}}(b) = \frac{1}{2}\mathbb{E}_{t\sim U[0,1]}\left\|b_t(X_t^{\bar{b}}) - b_t^{\text{ref}}(X_t^{\bar{b}})\right\|^2. \tag{10}$$

We combine the simplified reward and regularization losses, $\mathcal{L}_{\text{simple-reward}}$ and $\mathcal{L}_{\text{simple-regularization}}$. Since both simplified losses are regression problems at stopgradient samples, we may combine their targets into a single objective (§A.4), leading to the SIMPLE ENDPOINT REWARD FINETUNING objective for the drift b:

$$\mathcal{L}_{\text{SERF}}(b) = \mathbb{E}_{t\sim U[0,1]}\left\|b_t(X_t^{\bar{b}}) - b_t^{\text{ref}}(X_t^{\bar{b}}) - \nabla R(X_1^{\bar{b}})\right\|^2. \tag{11}$$

Sampling through renoising. Computing this loss requires sampling pairs (X_t^b, X_1^b) from the joint distribution $p_{t,1}^b$ induced by the SDE, which is possible through sampling X_1^b using either the SDE or associated ODE, and picking X_t^b according to $\text{Law}(X_t^b \mid X_1^b = x) = p_{t|1}^b$. When $b = b^\star$ is optimal for the SOC problem, Equation (4) gives $\text{Law}(X_t^{b^\star} \mid X_1^{b^\star} = x) = \text{Law}(X_t^{\text{ref}} \mid X_1^{\text{ref}} = x) = \mathcal{N}\big(tx, (1-t)^2 I\big)$. Thus, the conditional distribution is a Gaussian corruption kernel. We use this kernel for every b as a tractable proxy for $p_{t|1}^b$; it is exact both at initialization, $b = b^{\text{ref}}$, and at the optimum, $b = b^\star$. This Gaussian re-noising construction is also used in recent fine-tuning methods for non-differentiable rewards (Zheng et al., 2026b; Bergmeister et al., 2026).

Alternative methods (Lee et al., 2026; Liu et al., 2025b; Domingo-Enrich et al., 2025) instead sample from $p_{t,1}^b$ by simulating the forward process and choosing steps along this trajectory. For ODE sampling, this produces couplings (X_t, X_1) in which X_t is deterministic conditional on X_1, and thus poorly approximates the true conditional distribution $p_{t|1}^b$. SDE sampling recovers the correct marginals, but intermediate states drawn from the same trajectory yield correlated gradient estimates. We show that independently sampling intermediate states from the corruption kernel yields an estimator with strictly smaller gradient variance.

Proposition 2. *Assume* $\text{Law}(X_t^b \mid X_1^b) = \mathcal{N}(tX_1^b, (1-t)^2 I)$. *Define the single-sample gradients* $g_t(X_t, X_1) = \frac{1}{2}\nabla_\theta \|b_t(X_t) - b_t(X_t) - \nabla R(X_1)\|^2$. *For IID* $t_i \sim U[0,1]$, *consider the two estimates:*

- $g_{\text{traj}} = \frac{1}{m}\sum_{i=1}^m g_{t_i}(X_{t_i}, X_1)$ *where* $(X_t)_{t\in[0,1]}$ *is a trajectory under* b.
- $g_{\text{corr}} = \frac{1}{m}\sum_{i=1}^m g_{t_i}(X'_{t_i}, X_1)$ *where* $X_1 \sim p_1^b$ *and* $(X'_{t_i} \mid X_1, t_i) \sim \mathcal{N}(t_i X_1, (1-t_i)^2 I)$ *independently.*

Then g_{traj} *and* g_{corr} *are unbiased estimators of* $\nabla_\theta \mathcal{L}_{\text{simple-reward}}$, *and* $\text{Cov}(g_{\text{corr}}) \preceq \text{Cov}(g_{\text{traj}})$.

Instead of formulating a loss for the SDE drift, we may instead train the ODE velocity v directly. Under the assumption that both b and b^{ref} have the same backwards drift, which is true at the initialization and at $b^\star$, Equation (2) implies $b_t - b_t^{\text{ref}} = 2(v_t - v_t^{\text{ref}})$. Substituting this gives us the SERF objective for the velocity v:

$$\mathcal{L}_{\text{SERF}}(v) = \mathbb{E}_{t\sim U[0,1]}\left\|v_t(X_t^b) - v_t^{\text{ref}}(X_t^b) - \tfrac{1}{2}\nabla R(X_1^b)\right\|^2. \tag{12}$$

Combining this with a re-noising based sampler yield Equation (5).

4 EXPERIMENTS

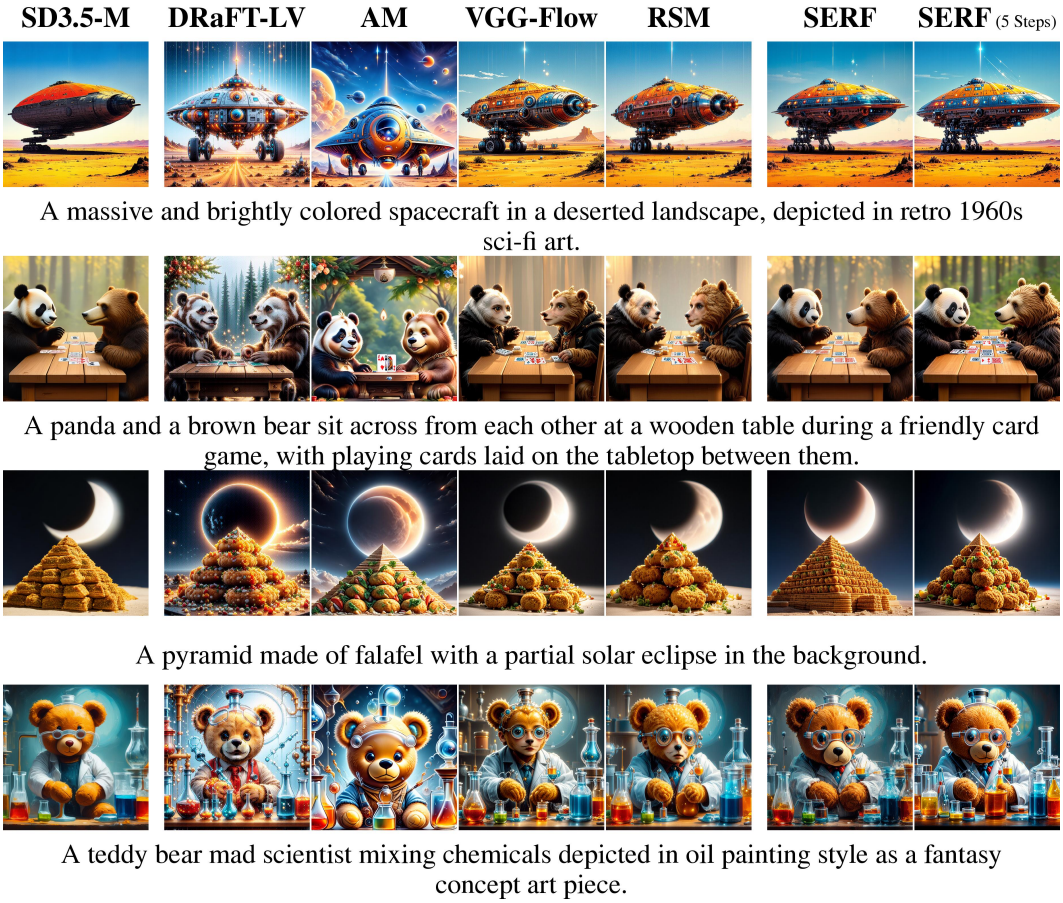

Figure 2: Qualitative comparison across five different prompts. The final column shows samples from SERF post-trained using five sampling steps per training rollout.

4.1 EXPERIMENTAL SETTINGS

Setup. We post-train Stable Diffusion 3.5 Medium (SD3.5-M) (Esser et al., 2024), a T2I flow matching model, and generate images at 512×512 resolution. We optimize two differentiable human-preference rewards: PickScore (Kirstain et al., 2023) and Human Preference Score v2.1 (HPSv2.1) (Wu et al., 2023). Training prompts come from Pick-a-Pic (Kirstain et al., 2023) for PickScore and Human Preference Dataset v2 (HPDv2) (Wu et al., 2023) for HPSv2.1.

Baselines. We compare against four[1] first-order methods that differ in how they use an endpoint reward to train the generative model. *VGG-Flow* (Liu et al., 2025b) matches the residual velocity to a value gradient formed from the reward gradient at a one-step clean-sample prediction and a learned correction. *Reward Score Matching* (RSM) (Lee et al., 2026) simplifies VGG-Flow's training procedure by removing auxiliary components, including the error correction network, and improves reward optimization at lower computational cost. *Adjoint Matching* (AM) (Domingo-Enrich et al., 2025) solves a backward adjoint equation along the sampled trajectory to construct intermediate training targets (§A). *DRaFT-LV* (Clark et al., 2024) backpropagates the reward through the final denoising step and averages gradients over multiple independently re-noised copies of each generated image.

Training Settings. We fine-tune SD3.5-M with LoRA (Hu et al., 2021) adapters of rank 32 and scaling parameter 64. All methods train for 300 gradient updates with AdamW (Loshchilov & Hutter, 2017), using $\beta_1 = 0.9$, $\beta_2 = 0.99$, and weight decay 10^{-4}. Each update uses 64 generated trajectories and four training targets per trajectory, for 256 targets in total. Generation during training

[1]We omit Efficient Adjoint Matching because no official implementation was available at the time of our experiments, and our reproduction attempts did not recover the reported results.

uses 10 Euler steps, except for Adjoint Matching, which uses 10 steps of its stochastic memoryless sampler (Domingo-Enrich et al., 2025). All methods use classifier-free guidance (CFG) of 4.5 and `bfloat16` precision. We maintain an exponential moving average (EMA) of the LoRA parameters with decay 0.9 and train on four NVIDIA GH200 GPUs. We tune the learning rate for DRaFT-LV and the reward coefficient β for the other methods separately for each task. The full search spaces are given in the appendix.

Evaluation Settings. Every 20 updates, we evaluate the EMA model using 20 Euler steps and CFG 4.5. For each task, we use 200 held-out prompts from its corresponding dataset and 200 DrawBench prompts (Saharia et al., 2022). We generate 16 images per prompt with fixed seeds, yielding 3,200 images per evaluation set. We report PickScore, HPSv2.1, ImageReward (Xu et al., 2023), and CLIPScore (Hessel et al., 2021). We measure within-prompt diversity as the mean pairwise cosine distance between DreamSim features (Fu et al., 2023) of the 16 images, averaged across prompts. We assess distribution shift using the Fréchet Inception Distance (FID) (Heusel et al., 2017) against images generated by the frozen base model with the same prompts.

4.2 Main Results

Quantitative Results. Figure 3 compares reward–quality trade-offs between SERF and the baselines, each trained with HPSv2.1 at five reward strengths. Highlighted curves show the run with the best overall trade-off for each method. SERF outperforms both RSM and VGG-Flow on T2I human-preference scores, image–prompt similarity, and FID, while maintaining comparable diversity. DRaFT-LV provides a weaker trade-off: although it strongly improves its training reward, the remaining metrics improve little. On the other hand, Adjoint Matching does not reach high HPSv2.1 rewards, even after 300 gradient updates. However, it remains competitive on ImageReward. This mismatch is also reflected in the qualitative results in Figure 2, where its generations appear less natural.

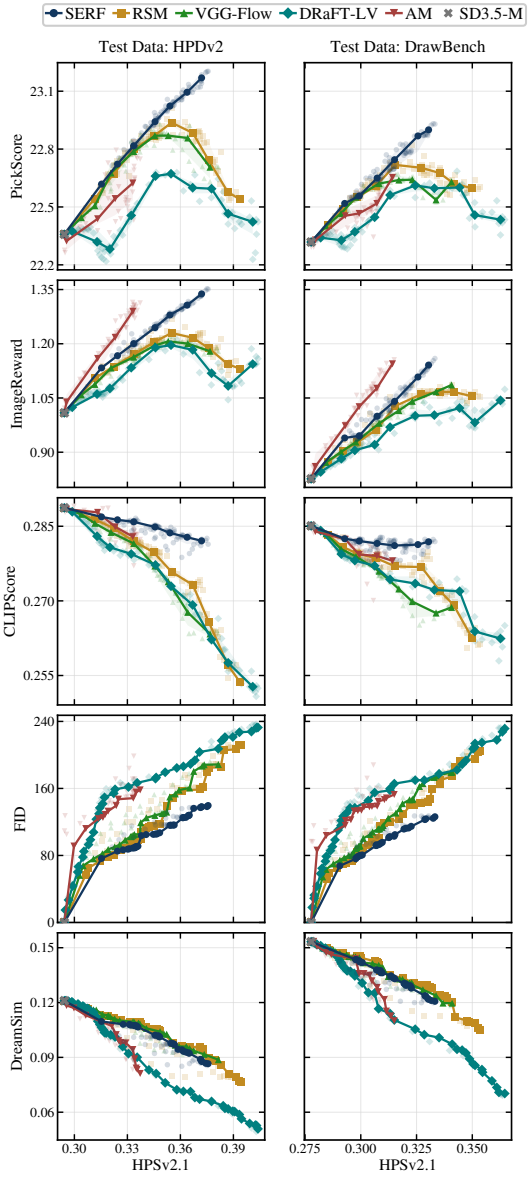

Figure 3: Reward–quality trade-offs for HPSv2.1-trained models, evaluated on HPDv2 (left) and DrawBench (right).

We additionally report reward and distribution metrics for models trained on PickScore and HPSv2.1 in Tables 1 and 2, respectively. We choose target reward levels intended to capture substantial gains without pronounced reward hacking, and compare checkpoints with similar values of the optimized reward. This allows us to assess the remaining metrics without favoring methods that achieve higher training rewards. Adjoint Matching does not reach the HPSv2.1 target, so we report its highest-reward checkpoint, despite clear signs of reward hacking and poor qualitative results (see Figure 2). VGG-Flow and Adjoint Matching also require 5–15$\times$ more training time, as detailed in §4.3. Nevertheless, across both tables, SERF achieves strong performance, ranking first or second on nearly every metric.

Qualitative Results. Figure 2 compares images generated by SD3.5-M, DRaFT-LV, Adjoint Matching, VGG-Flow, RSM, and two variants of SERF on five fixed prompts. We use the HPSv2.1-trained

Table 1: Comparison of PickScore-trained models. Selected checkpoints target a PickScore of 23.45 on Pick-a-Pic and 23.40 on DrawBench. Shaded cells denote the optimized reward.

Method	PickScore ↑	HPSv2 ↑	ImageReward ↑	CLIPScore ↑	FID ↓	Diversity ↑
Evaluation: Pick-a-Pic prompts						
SD3.5	19.840	0.207	−0.532	0.247	–	0.248
SD3.5 + CFG	21.543	0.293	0.971	0.290	–	0.128
DRaFT-LV	23.418 ± 0.024	0.3227 ± 0.0007	1.039 ± 0.009	0.2742 ± 0.0010	221.3 ± 4.4	0.0627 ± 0.0008
VGGFlow	23.367 ± 0.009	0.3149 ± 0.0011	1.088 ± 0.014	0.2836 ± 0.0007	180.7 ± 1.3	**0.1024 ± 0.0013**
Adjoint Matching	23.307 ± 0.080	**0.3353 ± 0.0013**	**1.426 ± 0.030**	0.2929 ± 0.0019	191.6 ± 7.0	0.0588 ± 0.0011
RSM	23.422 ± 0.031	0.3207 ± 0.0005	1.154 ± 0.012	0.2850 ± 0.0008	172.6 ± 5.4	0.0974 ± 0.0011
SERF	23.454 ± 0.020	0.3242 ± 0.0007	1.213 ± 0.003	**0.2969 ± 0.0002**	**144.9 ± 1.3**	0.0954 ± 0.0009
Evaluation: DrawBench prompts						
SD3.5	20.628	0.200	−0.611	0.239	–	0.279
SD3.5 + CFG	22.318	0.278	0.826	0.285	–	0.153
DRaFT-LV	23.416 ± 0.030	0.2983 ± 0.0006	0.880 ± 0.021	0.2645 ± 0.0002	190.7 ± 7.2	0.0918 ± 0.0015
VGGFlow	23.207 ± 0.012	0.2882 ± 0.0006	0.953 ± 0.020	0.2734 ± 0.0006	151.5 ± 1.9	**0.1328 ± 0.0005**
Adjoint Matching	23.625 ± 0.110	**0.3117 ± 0.0018**	**1.283 ± 0.030**	**0.2799 ± 0.0049**	168.6 ± 8.3	0.0823 ± 0.0032
RSM	23.332 ± 0.035	0.2939 ± 0.0006	1.018 ± 0.017	0.2751 ± 0.0017	148.0 ± 3.7	0.1271 ± 0.0012
SERF	23.366 ± 0.032	0.2977 ± 0.0005	1.081 ± 0.012	**0.2840 ± 0.0006**	**121.1 ± 1.6**	0.1284 ± 0.0003

Table 2: Comparison of HPSv2-trained models. Selected checkpoints target an HPSv2 score of 0.365 on HPSv2 prompts and 0.325 on DrawBench. Shaded cells denote the optimized reward.

Method	PickScore ↑	HPSv2 ↑	ImageReward ↑	CLIPScore ↑	FID ↓	Diversity ↑
Evaluation: HPDv2 prompts						
SD3.5	20.517	0.206	−0.357	0.250	–	0.241
SD3.5 + CFG	22.358	0.294	1.009	0.289	–	0.121
DRaFT-LV	22.700 ± 0.098	0.3648 ± 0.0029	1.214 ± 0.038	0.2682 ± 0.0035	193.0 ± 5.7	0.0711 ± 0.0021
VGGFlow	22.858 ± 0.057	0.3645 ± 0.0014	1.204 ± 0.008	0.2708 ± 0.0009	160.4 ± 0.2	**0.0967 ± 0.0018**
Adjoint Matching	22.593 ± 0.201	0.3382 ± 0.0029	**1.282 ± 0.086**	0.2755 ± 0.0044	189.1 ± 18.8	0.0744 ± 0.0067
RSM	22.882 ± 0.021	0.3653 ± 0.0021	1.232 ± 0.015	0.2753 ± 0.0004	148.0 ± 1.9	**0.0988 ± 0.0005**
SERF	**23.121 ± 0.012**	0.3639 ± 0.0007	**1.316 ± 0.006**	**0.2838 ± 0.0008**	**126.5 ± 1.1**	0.0924 ± 0.0009
Evaluation: DrawBench prompts						
SD3.5	20.628	0.200	−0.611	0.239	–	0.279
SD3.5 + CFG	22.318	0.278	0.826	0.285	–	0.153
DRaFT-LV	22.639 ± 0.035	0.3277 ± 0.0028	1.032 ± 0.006	0.2709 ± 0.0024	174.3 ± 6.1	0.1039 ± 0.0029
VGGFlow	22.632 ± 0.025	0.3231 ± 0.0014	1.025 ± 0.011	0.2724 ± 0.0027	145.5 ± 1.6	**0.1316 ± 0.0022**
Adjoint Matching	22.508 ± 0.103	0.3127 ± 0.0039	**1.110 ± 0.034**	0.2717 ± 0.0058	178.0 ± 19.6	0.1011 ± 0.0100
RSM	22.664 ± 0.012	0.3240 ± 0.0011	1.034 ± 0.010	0.2786 ± 0.0013	132.8 ± 1.0	**0.1340 ± 0.0004**
SERF	**22.785 ± 0.003**	0.3216 ± 0.0004	**1.084 ± 0.007**	**0.2825 ± 0.0004**	**114.0 ± 1.4**	0.1289 ± 0.0007

checkpoints reported in Table 2. DRaFT-LV and Adjoint Matching often produce unnatural images. VGG-Flow and RSM produce animals with human-like facial features for several prompts, an apparent reward hacking artifact. SERF produces natural and prompt-consistent images across the displayed prompts. The variant post-trained with only five sampling steps also produces high quality, text-aligned images, showing that SERF remains effective with short training trajectories. Additional qualitative results are provided in §C.3.

4.3 Ablations

Convergence speed. Figure 4 compares how quickly SERF, RSM, and Adjoint Matching improve HPSv2.1 during post-training. Adjoint Matching does not reach high HPSv2.1 rewards within 300 gradient updates and requires roughly 10× more training time than SERF to reach comparable improvement. SERF also improves faster than RSM. In our setting, SERF fine-tunes SD3.5-M in roughly three GPU hours.

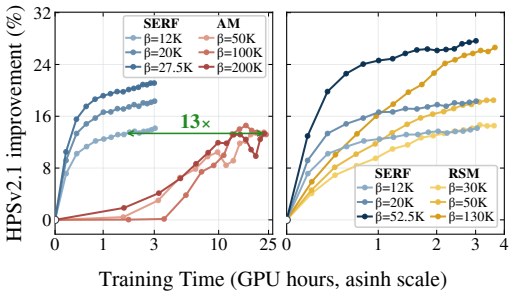

Figure 4: HPSv2.1 improvement over training time for SERF, AM (left), and RSM (right).

Training settings. Figure 5 studies two properties of SERF. The top row varies the CFG scale used during post-training (1, 2, 3, 4.5), while evaluation always uses CFG 4.5. Reducing the training CFG shifts the reward–quality trade-off, generally improving the reward metrics at the cost of lower diversity and changes in FID. The bottom row compares five and ten sampling steps per training rollout. SERF still obtains substantial reward improvements with only five steps. This is enabled by re-noising: after generating an endpoint, SERF draws independently noised intermediate training states, so the number of training targets is not limited by the number of states in the sampling rollout.

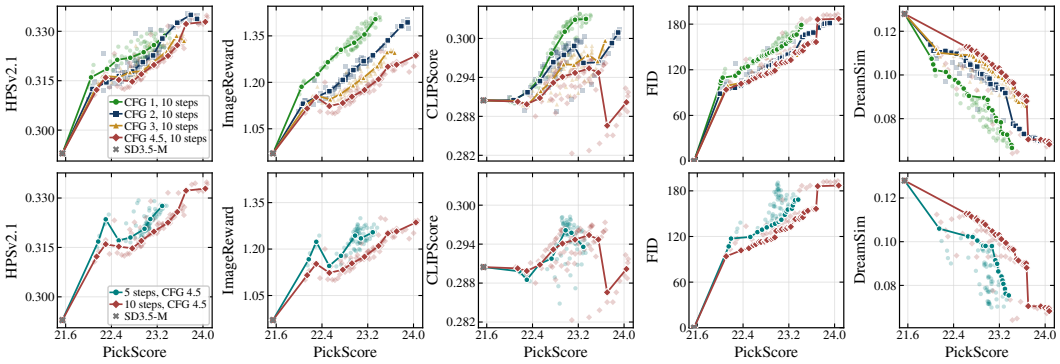

Figure 5: SERF trained on PickScore and evaluated on Pick-a-pic. Top: Comparison for different training CFG scales. Bottom: using five or ten sampling steps per training rollout.

5 Related Work

Reward finetuning methods for diffusion and flow models differ in whether they require differentiable rewards. We focus on training-time methods; see Uehara et al. (2025) for test-time approaches.

Non-differentiable rewards. *Trajectory* methods such as DDPO, DPOK, Flow-GRPO, and DanceGRPO apply policy gradients to Gaussian denoising transitions (Black et al., 2024; Liu et al., 2026; Fan et al., 2023; Xue et al., 2025). *ELBO* methods reward complete generations through pretraining-like losses: FPO (McAllister et al., 2025; Choi et al., 2026b) uses likelihood bounds, AWM (Xue et al., 2026) advantage weights, DiffusionNFT (Zheng et al., 2026b; Kveton et al., 2026; Fan et al., 2025) high- and low-reward samples, and RAM (Bergmeister et al., 2026) a stochastic-control target. FDFO (McAllister et al., 2026) applies an endpoint direction at intermediate states, estimated from paired generations without reward gradients.

Differentiable rewards. AlignProp backpropagates reward gradients through the sampling trajectory (Prabhudesai et al., 2023), while DRaFT-K, DRTune, and LeapAlign limit the backward computation through stop gradients, respectively (Clark et al., 2024; Wu et al., 2024; Liang et al., 2026). ReFL and SQDF use intermediate clean-image estimates for reward guidance (Xu et al., 2023; Kang et al., 2026). VGG-Flow learns an auxiliary guidance network (Liu et al., 2025a); Adjoint Matching integrates backward adjoint dynamics, while EAM obtains a closed-form adjoint but requires the pretrained terminal score (Domingo-Enrich et al., 2025; Shin et al., 2026). ∇-GFlowNet uses reward gradients in a detailed-balance objective (Liu et al., 2025c), whereas RSM applies endpoint reward gradients at sampled trajectory states (Lee et al., 2026). We discuss these connections in §B.

6 Conclusion

We introduce Simple Endpoint Reward Finetuning, a first-order method for reward finetuning flow models without backpropagating through the sampling trajectory. SERF computes the reward gradient at a generated endpoint, independently re-noises that endpoint to obtain intermediate training states, and regresses the velocity toward the reference velocity plus the endpoint guidance. We show that this guidance aligns with the exact reward guidance under our stated assumptions, and that independent re-noising reduces the variance of training targets. When fine-tuning Stable Diffusion 3.5 on PickScore and HPSv2.1, SERF achieves strong trade-offs among reward, image quality, and diversity, while remaining effective with short sampling rollouts. Its simplicity comes from approximating the path conditional Jacobian with the identity. This motivates studying inexpensive corrections to that approximation that preserve the efficiency of endpoint re-noising.

AI USE STATEMENT

In this work, we used generative AI tools to improve the clarity of the author-written text, including the presentation of mathematical claims, identify potentially relevant literature, assist with implementation of the method, and provide critical feedback on our research ideas and methodology. We did not use generative AI tools to generate synthetic datasets. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This paper presents work whose goal is to advance the field of machine learning. There are many potential societal consequences of our work, none of which we feel needs to be specifically highlighted here.

REPRODUCIBILITY STATEMENT

We describe the SERF in the main text and provide implementation details, hyperparameters, and proofs in the appendix. All datasets used are publicly available. We will release our code in a public GitHub repository upon publication.

REFERENCES

Michael Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025. 1, 2

Fan Bao, Shen Nie, Kaiwen Xue, Chongxuan Li, Shi Pu, Yaole Wang, Gang Yue, Yue Cao, Hang Su, and Jun Zhu. One transformer fits all distributions in multi-modal diffusion at scale. In *International Conference on Machine Learning*, pp. 1692–1717. PMLR, 2023. 1

Andreas Bergmeister, Stefanie Jegelka, Nikolas Nüsken, Carles Domingo-Enrich, and Jakiw Pidstrigach. Reinforce adjoint matching: Scaling rl post-training of diffusion and flow-matching models, 2026. URL https://arxiv.org/abs/2605.10759. 1, 5, 9, 21

Julius Berner, Lorenz Richter, and Karen Ullrich. An optimal control perspective on diffusion-based generative modeling. *arXiv preprint arXiv:2211.01364*, 2022. 3, 5

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2305.13301. 9, 21

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22563–22575. IEEE, 2023. 1

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165. 1

Jaemoo Choi, Wei Guo, Yuchen Zhu, Arash Vahdat, Molei Tao, Julius Berner, and Yongxin Chen. Scaling reinforcement learning for diffusion models via velocity matching, 2026a. URL https://arxiv.org/abs/2608.23664. 5

Jaemoo Choi, Yuchen Zhu, Wei Guo, Petr Molodyk, Bo Yuan, Jinbin Bai, Yi Xin, Molei Tao, and Yongxin Chen. Rethinking the design space of reinforcement learning for diffusion models: On the importance of likelihood estimation beyond loss design, 2026b. 9

Kevin Clark, Paul Vicol, Kevin Swersky, and David J. Fleet. Directly fine-tuning diffusion models on differentiable rewards. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2309.17400. 1, 6, 9, 21

Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2409.08861. 1, 2, 3, 4, 5, 6, 7, 9, 21

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. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2403.03206. 1, 2, 6

Jiajun Fan, Shuaike Shen, Chaoran Cheng, Yuxin Chen, Chumeng Liang, and Ge Liu. Online reward-weighted fine-tuning of flow matching with wasserstein regularization, 2025. 9, 21

Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. DPOK: Reinforcement learning for fine-tuning text-to-image diffusion models. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2305.16381. 9, 21

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=OlzB6LnXcS. 1

Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. Dreamsim: Learning new dimensions of human visual similarity using synthetic data. *arXiv preprint arXiv:2306.09344*, 2023. 7

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=uWj4s7rMnR. 1

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 7514–7528, 2021. 7

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. 7

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020a. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf. 5

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020b. URL https://arxiv.org/abs/2006.11239. 1

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. 1

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 6

Zhiwei Jia, Yuesong Nan, Huixi Zhao, and Gengdai Liu. Reward fine-tuning two-step diffusion models via learning differentiable latent-space surrogate reward. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12912–12922. IEEE, 2025. 2

Hyeongyu Kang, Jaewoo Lee, Woocheol Shin, Kiyoung Om, and Jinkyoo Park. Diffusion fine-tuning via reparameterized policy gradient of the soft q-function. In *International Conference on Learning Representations*, volume 2026, pp. 78690–78725, 2026. 2, 9, 21

Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. Pick-a-pic: An open dataset of user preferences for text-to-image generation. *arXiv preprint arXiv:2305.01569*, 2023. URL https://arxiv.org/abs/2305.01569. 2, 6

Hiroshi Kunita. *Stochastic Flows and Stochastic Differential Equations*, volume 24 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, 1990. 17

Branislav Kveton, Anup Rao, Subhojyoti Mukherjee, Krishna Kumar Singh, and Viet Dac Lai. AdvantageFlow: Advantage-weighted least squares for RL in flow models, 2026. 9

Jeongjae Lee, Jinho Chang, Jeongsol Kim, and Jong Chul Ye. Reward score matching: Unifying reward-based fine-tuning for flow and diffusion models. *arXiv preprint arXiv:2604.17415*, 2026. 5, 6, 9, 21

Zhanhao Liang, Tao Yang, Jie Wu, Chengjian Feng, and Liang Zheng. LeapAlign: Post-training flow matching models at any generation step by building two-step trajectories, 2026. 9

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.02747. 1, 2

Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D Plumbley. Audioldm: Text-to-audio generation with latent diffusion models. *arXiv preprint arXiv:2301.12503*, 2023. 1

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online rl. *Advances in neural information processing systems*, 38:40783–40818, 2026. 9, 21

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022. 1

Zhen Liu, Tim Z. Xiao, Carles Domingo-Enrich, Weiyang Liu, and Dinghuai Zhang. Value gradient guidance for flow matching alignment. In *Advances in Neural Information Processing Systems*, 2025a. 2, 9, 21

Zhen Liu, Tim Z. Xiao, Carles Domingo-Enrich, Weiyang Liu, and Dinghuai Zhang. Value gradient guidance for flow matching alignment. In *Advances in Neural Information Processing Systems*, 2025b. URL https://arxiv.org/abs/2512.05116. 5, 6

Zhen Liu, Tim Z. Xiao, Weiyang Liu, Yoshua Bengio, and Dinghuai Zhang. Efficient diversity-preserving diffusion alignment via gradient-informed GFlowNets, 2025c. 9

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

David McAllister, Songwei Ge, Brent Yi, Chung Min Kim, Ethan Weber, Hongsuk Choi, Haiwen Feng, and Angjoo Kanazawa. Flow matching policy gradients. *arXiv preprint arXiv:2507.21053*, 2025. 9, 21

David McAllister, Miika Aittala, Tero Karras, Janne Hellsten, Angjoo Kanazawa, Timo Aila, and Samuli Laine. Finite difference flow optimization for RL post-training of text-to-image models. *arXiv preprint arXiv:2603.12893*, 2026. URL https://arxiv.org/abs/2603.12893. 4, 9, 21

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022. 1

Stefano Peluchetti. Non-denoising forward-time diffusions, 2021. 1

Mihir Prabhudesai, Anirudh Goyal, Deepak Pathak, and Katerina Fragkiadaki. Aligning text-to-image diffusion models with reward backpropagation. 2023. 1, 9, 21

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10674–10685. ieee, 2022. 1

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022. 7

Jeongwoo Shin, Dongsoo Shin, Yuchen Zhu, Wei Guo, Yongxin Chen, Joonseok Lee, Jaewoong Choi, and Jaemoo Choi. Efficient adjoint matching for fine-tuning diffusion models. *arXiv preprint arXiv:2605.11480*, 2026. URL https://arxiv.org/abs/2605.11480. 1, 9, 21

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2011.13456. 1, 2

Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research (TMLR)*, 2024. 1

Masatoshi Uehara, Yulai Zhao, Chenyu Wang, Xiner Li, Aviv Regev, Sergey Levine, and Tommaso Biancalani. Inference-time alignment in diffusion models with reward-guided generation: Tutorial and review. *arXiv preprint arXiv:2501.09685*, 2025. 9

Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. *arXiv preprint arXiv:2306.09341*, 2023. URL https://arxiv.org/abs/2306.09341. 2, 6

Xiaoshi Wu, Yiming Hao, Manyuan Zhang, Keqiang Sun, Zhaoyang Huang, Guanglu Song, Yu Liu, and Hongsheng Li. Deep reward supervisions for tuning text-to-image diffusion models. In *European Conference on Computer Vision*, pp. 108–124. Springer, 2024. 9

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. In *Advances in Neural Information Processing Systems*, volume 36, 2023. 1, 2, 7, 9, 21

Shuchen Xue, Chongjian GE, Shilong Zhang, Yichen Li, and Zhi-Ming Ma. Advantage weighted matching: Aligning RL with pretraining in diffusion models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=nLY2pOYBrJ. 5, 9

Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, et al. Dancegrpo: Unleashing grpo on visual generation. *arXiv preprint arXiv:2505.07818*, 2025. 9, 21

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. Diffusionnft: Online diffusion reinforcement with forward process. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 134129–134150, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/d8e68ddfe22520b45b8fb8d5cbde5a21-Paper-Conference.pdf. 5

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. Diffusionnft: Online diffusion reinforcement with forward process. In *International Conference on Learning Representations*, volume 2026, pp. 134129–134150, 2026b. 5, 9

Zhiyuan Zhou, Andy Peng, Charles Xu, Qiyang Li, Jost Tobias Springenberg, Kevin Frans, and Sergey Levine. Test-time gradient guidance of flow policies in reinforcement learning, 2026. 4

Appendix

A Proofs **16**

A.1 Stochastic optimal control proofs 16

A.2 The stochastic Jacobian 17

A.3 Derivation of the simple endpoint estimate 18

A.4 Combining regression targets 19

B Extended Related Work **21**

C Additional Results **22**

C.1 Quantitative Results 22

C.2 Ablations 23

C.3 Qualitative Results 26

A PROOFS

A.1 STOCHASTIC OPTIMAL CONTROL PROOFS

For a diffusion process satisfying

$$\mathrm{d}X_t^b = b_t(X_t)\mathrm{d}t + \sigma_t \mathrm{d}W_t, \qquad X_0^b \sim p_0, \tag{13}$$

consider the optimization of a loss of the form

$$J(\theta) = \mathbb{E}\left[\int_0^1 L(X_s^b, b_s, s)\mathrm{d}s + R(X_1^b)\right], \tag{14}$$

where θ are the parameters underlying the diffusion drift b_t. Introducing the value function

$$V(x,t) = \mathbb{E}\left[\int_t^1 L(X_s^b, b_s, s)\mathrm{d}s + R(X_1^b) \mid X_t = x\right], \tag{15}$$

we obtain the following:

Theorem 2. *The gradient of the reward J can be exactly written as*

$$\nabla_\theta J(\theta) = \mathbb{E}\left[\int_0^1 \nabla_\theta b_t(X_t^b)^\top (\nabla_b L(X_t^b, b_t, t) + \nabla_x V(X_t^b, t))\mathrm{d}t\right]. \tag{16}$$

Hence, maximizing J through gradient ascent corresponds to minimizing the following surrogate objective through gradient descent

$$\hat{\mathcal{L}}(\theta) = \frac{1}{2}\mathbb{E}\left[\int_0^1 \left\| b_t(X_t^{\bar{b}}) - \bar{b}_t(X_t^{\bar{b}}) - \nabla_b L(X_t^{\bar{b}}, \bar{b}_t, t) - \nabla_x V(X_t^{\bar{b}}, t)\right\|^2\right]. \tag{17}$$

Proof. The idea is to use the Feynman-Kac equation twice, the first to obtain a PDE which V satisfies, and the second to go from a PDE that $\nabla_\theta V$ satisfies to an equation for $\nabla_\theta V$. Firstly, from the definition of the value function Equation (15) and the form of the SDE Equation (13), Feynman-Kac tells us that

$$\frac{\partial}{\partial t}V(x,t) + b_t(x)\nabla_x V(x,t) + \frac{1}{2}\sigma_t^2 \Delta V(x,t) + L(x, b_t(x), t) = 0, \tag{18}$$

with terminal condition $V(x,1) = R(x)$. We take the gradient of this equation with respect to θ. Let $\nabla_\theta V = U$. Then, we obtain

$$\frac{\partial}{\partial t}U(x,t) + \nabla_\theta b_t(x)^\top \nabla_x V(x,t) + b_t(x)\nabla_x U(x,t) \tag{19}$$

$$+ \frac{1}{2}\sigma_t^2 \Delta U(x,t) + \nabla_\theta b_t(x)^\top \nabla_b L(x, b_t(x), t) = 0. \tag{20}$$

Moreover, $U(x,1) = \nabla_\theta V(x,1) = \nabla_\theta R(x) = 0$. From Feynman-Kac, this means we can express U as exactly

$$U(x,t) = \mathbb{E}\left[\int_t^1 \nabla_\theta b_s(X_s^b)^\top (\nabla_b L(X_s^b, b_s, s) + \nabla_x V(X_s^b, s))\mathrm{d}s \mid X_t = x\right]. \tag{21}$$

Then, we can represent $J(\theta) = \mathbb{E}_{X_0 \sim p_0}[V(X_0, 0)]$, so since p_0 is independent of θ, $\nabla_\theta J(\theta) = \mathbb{E}_{X_0 \sim p_0}[U(X_0, 0)]$ which gives exactly Equation (16) by plugging in the above expression for U.

To show that the gradient of the surrogate loss $\hat{\mathcal{L}}$ is the negation of $\nabla_\theta J(\theta)$, we simply differentiate $\hat{\mathcal{L}}$. Since sampling is done under a stop gradient distribution, we can simply differentiate the interior of the integral, which is

$$\nabla_\theta \tfrac{1}{2}\| b_t(X_t^{\bar{b}}) - \bar{b}_t(X_t^{\bar{b}}) - \nabla_b L(X_t^{\bar{b}}, \bar{b}_t, t) - \nabla_x V(X_t^{\bar{b}}, t)\|^2 \tag{22}$$

$$= \nabla_\theta b_t(X_t^b)^\top (b_t(X_t^b) - b_t(X_t^b) - \nabla_b L(X_t^b, b_t, t) - \nabla_x V(X_t^b, t)) \tag{23}$$

$$= -\nabla_\theta b_t(X_t^b)^\top (\nabla_b L(X_t^b, b_t, t) + \nabla_x V(X_t^b, t). \tag{24}$$

Integrating this yields $\nabla_\theta \hat{\mathcal{L}}(\theta) = -\nabla_\theta J(\theta)$. □

Taking $L = 0$ in Theorem 2 yields exactly Theorem 1.

Moreover, we may derive Adjoint Matching from this formula. In Adjoint Matching, we take $L = -\|b_t - b_t^{\text{ref}}\|^2/(2\sigma(t)^2)$, which when substituted into Theorem 2 gives

$$\mathbb{E}_{\theta^-,t}\left\|\frac{(b_t(X_t^{\bar{b}}) - b_t^{\text{ref}}(X_t^{\bar{b}}))}{\sigma(t)^2} - \nabla_x V(X_t^{\bar{b}}, t)\right\|^2 \tag{25}$$

This is equivalent to the basic adjoint matching loss, up to reweighting.

A.2 The stochastic Jacobian

Assume throughout that $\bar{b} \in C([0,1]; C_b^2(\mathbb{R}^d;\mathbb{R}^d))$, $R \in C_b^1(\mathbb{R}^d)$, and σ_s is continuous and independent of the state. Under these assumptions, the SDE generates a C^1 stochastic flow (Kunita, 1990, Chapter 4).

Given a realization ξ of the Brownian motion and $X_t = x$, define

$$T_{s,t}^{\xi}(x) := X_s^{t,x,\bar{b}}(\xi), \qquad s \in [t,1].$$

For fixed ξ, $T_{s,t}^{\xi}$ is a deterministic function of x. Its stochastic Jacobian is

$$J_{s,t}^{\xi}(x) := \nabla_x T_{s,t}^{\xi}(x).$$

Since σ_s is independent of the state, the stochastic Jacobian satisfies

$$\frac{\mathrm{d}}{\mathrm{d}s} J_{s,t}^{\xi}(x) = \nabla \bar{b}_s\left(T_{s,t}^{\xi}(x)\right) J_{s,t}^{\xi}(x), \qquad J_{t,t}^{\xi}(x) = I.$$

Differentiation and expectation can be interchanged under the assumptions above. Applying the chain rule pathwise gives

$$\nabla_x \mathbb{E}\left[R(X_1^{\bar{b}}) \mid X_t^{\bar{b}} = x\right] = \mathbb{E}_{\xi}\left[J_{1,t}^{\xi}(x)^{\top} \nabla R\left(T_{1,t}^{\xi}(x)\right)\right].$$

We can furthermore show that the endpoint Jacobian approaches the identity as $t \to 1$.

Proposition 3. *Under the assumptions above,*

$$J_{1,t}^{\xi}(x) = I + \mathcal{O}(1-t)$$

almost surely.

Proof. The differential equation for the stochastic Jacobian implies

$$J_{s,t}^{\xi}(x) = I + \int_t^s \nabla \bar{b}_r\left(T_{r,t}^{\xi}(x)\right) J_{r,t}^{\xi}(x)\,\mathrm{d}r.$$

Since $\bar{b} \in C([0,1]; C_b^2)$, there exists $L < \infty$ such that

$$\sup_{r,x} \|\nabla \bar{b}_r(x)\|_{\text{op}} \le L.$$

Consequently,

$$\|J_{1,t}^{\xi}(x) - I\|_{\text{op}} \le L \int_t^1 \|J_{r,t}^{\xi}(x)\|_{\text{op}}\,\mathrm{d}r \le e^{L(1-t)} - 1 = \mathcal{O}(1-t),$$

which proves the result. □

A.3 Derivation of the simple endpoint estimate

Proposition 4. *The simple endpoint estimate for* $\nabla_x \mathbb{E}[R(X_1) \mid X_t = x]$ *is*

$$\hat{R}_t(x) = \mathbb{E}[\nabla R(X_1) \mid X_t = x]. \tag{26}$$

$\hat{R}_t(x)$ *is positively aligned with* $\nabla_x \mathbb{E}[R(X_1) \mid X_t = x]$ *if* $\mathrm{Law}(X_1 \mid X_t = x) = \mathcal{N}(T(x), \sigma^2 I)$, *where* $J_T(x) = \nabla_x T(x)$ *has positive-definite symmetric part. In particular, if* $\mathrm{Law}(X_t \mid X_1 = x)$ *is Gaussian, then this holds for* $T(x) = \mathbb{E}[X_1 \mid X_t = x]$.

Proof. If $\mathrm{Law}(X_1 \mid X_t = x) = \mathcal{N}(T(x), \sigma^2 I)$, then we may write $X_1 = T(X_t) + \sigma Z$ for $Z \sim \mathcal{N}(0, I)$. Thus,

$$\nabla_x \mathbb{E}[R(X_1) \mid X_t = x] = \nabla_x \mathbb{E}[R(T(x) + \sigma z)] = \mathbb{E}[\nabla_x T(x)^\top \nabla_x R(T(x) + \sigma z)] \tag{27}$$

$$= J_T(x)^\top \mathbb{E}[\nabla R(X_1) \mid X_t = x]. \tag{28}$$

This is exactly $\mathbb{E}[\nabla R(X_1) \mid X_t = x]$ when $J_T(x) = I$, i.e. $T(x)$ is locally a shift. In particular, when $T(x) = x + c$ then the simple endpoint estimate is exact for all x.

In general, when $J_T(x) \neq I$, we can still calculate when the simple endpoint estimate is positively aligned with $\nabla_x \mathbb{E}[R(X_1) \mid X_t = x]$. The inner product between the two is

$$\mathbb{E}[\nabla R(X_1) \mid X_t = x]^\top J_T(x)^\top \mathbb{E}[R(X_1) \mid X_t = x] \tag{29}$$

$$= \mathbb{E}[\nabla R(X_1) \mid X_t = x]^\top \left(\frac{J_T(x) + J_T(x)^\top}{2} \right) \mathbb{E}[\nabla R(X_1) \mid X_t = x] \geq 0 \tag{30}$$

when $(J_T + J_T^\top)/2$ is positive definite, showing the first part.

We now proceed to show that J_T has positive definite symmetric part if $T(x) = \mathbb{E}[X_1 \mid X_t = x]$, under Gaussian backwards law. Indeed, under the assumption $\mathrm{Law}(X_t \mid X_1) = \mathcal{N}(aX_1, \sigma^2 I)$, Tweedie's gives that $\nabla_x \log p_{1|t}(y|x) = \frac{a}{\sigma^2}(y - T(x))$. This gives

$$\nabla_x \mathbb{E}[X_1 \mid X_t = x] = \nabla_x \int y p_{1|t}(y|x) \mathrm{d}y \tag{31}$$

$$= \frac{a}{\sigma^2} \int y(y - T(x)) p_{1|t}(y|x) \mathrm{d}y \tag{32}$$

$$= \frac{a}{\sigma^2} \int (y - T(x))(y - T(x)) p_{1|t}(y|x) \mathrm{d}y \tag{33}$$

$$= \frac{a}{\sigma^2} \mathrm{Cov}(X_1 - T(x) \mid X_t = x) \succeq 0. \tag{34}$$

□

If we plug in the simple endpoint estimate into $\hat{\mathcal{L}}_{\text{reward}}$, we obtain the loss

$$\mathcal{L}_{\text{simple-expected-reward}} = \frac{1}{2} \mathbb{E}_{t \sim U[0,1]} \left\| b_t(X_t^{\bar{b}}) - \bar{b}_t(X_t^{\bar{b}}) - \mathbb{E}[\nabla R(X_1^{\bar{b}}) \mid X_t^{\bar{b}}] \right\|^2. \tag{35}$$

Lemma 1. *The losses* $\mathcal{L}_{\text{simple-expected-reward}}$ *and* $\mathcal{L}_{\text{simple-reward}}$ *from Equation (9) have the same gradients.*

Proof. We rearrange the losses:

$$\mathcal{L}_{\text{simple-expected-reward}} \tag{36}$$

$$= \frac{1}{2} \mathbb{E}_{t \sim U[0,1]} \left\| b_t(X_t^{\bar{b}}) - \bar{b}_t(X_t^{\bar{b}}) - \nabla R(X_1^{\bar{b}}) + (\nabla R(X_1^{\bar{b}}) - \mathbb{E}[\nabla R(X_1^{\bar{b}}) \mid X_t^{\bar{b}}]) \right\|^2 \tag{37}$$

$$= \mathcal{L}_{\text{simple-reward}} \tag{38}$$

$$+ \mathbb{E}_{t \sim U[0,1]} \langle b_t(X_t^{\bar{b}}) - \bar{b}_t(X_t^{\bar{b}}) - \nabla R(X_1^{\bar{b}}), \nabla R(X_1^{\bar{b}}) - \mathbb{E}[\nabla R(X_1^{\bar{b}}) \mid X_t^{\bar{b}}] \rangle \tag{39}$$

$$+ \frac{1}{2} \mathbb{E}_{t \sim U[0,1]} \left\| \nabla R(X_1^{\bar{b}}) - \mathbb{E}[\nabla R(X_1^{\bar{b}}) \mid X_t^{\bar{b}}] \right\|^2. \tag{40}$$

The final term is constant in b since all terms contain a stop gradient, hence it does not contribute to the gradient. In the middle term, the only term without stop gradients is $\langle b_t(X_t^{\bar{b}}), \nabla R(X_1^{\bar{b}}) - \mathbb{E}[\nabla R(X_1^{\bar{b}}) \mid X_t^{\bar{b}}]\rangle$. However, its expectation is exactly 0, since conditioning with respect to $X_t^{\bar{b}}$ gives that this is equal to

$$\mathbb{E}\langle b_t(X_t^{\bar{b}}), \mathbb{E}[\nabla R(X_1^{\bar{b}}) - \mathbb{E}[\nabla R(X_1^{\bar{b}}) \mid X_t^{\bar{b}}] \mid X_t^{\bar{b}}]\rangle = 0, \tag{41}$$

since the right hand side of the inner product is simply 0. All other terms apart from $\mathcal{L}_{\text{simple}-\text{reward}}$ have stop gradient terms, so the gradients with respect to θ are the same. □

We now prove that under Proposition 1 optimizing Equation (9) increases the reward.

Proposition 5. *Under Proposition 1, minimizing Equation (9) yields a reward-ascent direction.*

Proof. Let

$$g_t(x) := \nabla_x \mathbb{E}[R(X_1) \mid X_t = x], \qquad \hat{R}_t(x) := \mathbb{E}[\nabla R(X_1) \mid X_t = x].$$

At $b = \bar{b}$, the negative gradient of Equation (9) with respect to $b_t(x)$ is $\hat{R}_t(x)$. The directional derivative of the expected reward along this update is

$$\frac{\mathrm{d}}{\mathrm{d}\eta}\mathcal{L}_{\text{reward}}(b + \eta\hat{R})\Big|_{\eta=0} = \mathbb{E}_{t,X_t}\left[\left\langle g_t(X_t), \hat{R}_t(X_t)\right\rangle\right].$$

By Proposition 1, the inner product is positive. Hence, for sufficiently small $\eta > 0$,

$$\mathcal{L}_{\text{reward}}(b + \eta\hat{R}) > \mathcal{L}_{\text{reward}}(b),$$

so minimizing Equation (9) performs reward ascent. □

A.4 Combining regression targets

Lemma 2. *Consider the combined loss*

$$\frac{1}{2}\mathbb{E}_{t\sim U[0,1]}\left\|b_t(X_t^{\bar{b}}) - \bar{b}_t(X_t^{\bar{b}}) - \nabla R(X_1^{\bar{b}})\right\|^2 + \frac{1}{2}\mathbb{E}_{t\sim U[0,1]}\left\|b_t(X_t^{\bar{b}}) - b_t^{\text{ref}}(X_t^{\bar{b}})\right\|^2. \tag{42}$$

Then this is equivalent to the SERF *loss*

$$\mathcal{L}_{\text{SERF}}(b) = \mathbb{E}_{t\sim U[0,1]}\left\|b_t(X_t^{\bar{b}}) - b_t^{\text{ref}}(X_t^{\bar{b}}) - \nabla R(X_1^{\bar{b}})\right\|^2. \tag{43}$$

Proof. The gradient with respect to the first is

$$\mathbb{E}_{t\sim U[0,1]}\nabla_\theta b_t(X_t^b)(-\nabla R(X_1^b)) + \mathbb{E}_{t\sim U[0,1]}\nabla_\theta b_t(X_t^b)(b_t(X_t^b) - b_t^{\text{ref}}(X_t^b)) \tag{44}$$

$$= \mathbb{E}_{t\sim U[0,1]}\nabla_\theta b_t(X_t^b)(b_t(X_t^b) - b_t^{\text{ref}}(X_t^b) - \nabla R(X_1^b)), \tag{45}$$

which is exactly (half) the gradient of the second objective. □

Proposition 6. *Assume* $\text{Law}(X_t^b \mid X_1^b) = \mathcal{N}(tX_1^b, (1-t)^2 I)$. *Define the single-sample gradients* $g_t(X_t, X_1) = \frac{1}{2}\nabla_\theta\|b_t(X_t) - \bar{b}_t(X_t) - \nabla R(X_1)\|^2$. *For IID* $t_i \sim U[0,1]$, *consider the two estimates:*

- $g_{\text{traj}} = \frac{1}{m}\sum_{i=1}^m g_{t_i}(X_{t_i}, X_1)$ *where* $(X_t)_{t\in[0,1]}$ *is a trajectory under* b.
- $g_{\text{corr}} = \frac{1}{m}\sum_{i=1}^m g_{t_i}(X'_{t_i}, X_1)$ *where* $X_1 \sim p_1^b$ *and* $(X'_{t_i} \mid X_1, t_i) \sim \mathcal{N}(t_i X_1, (1-t_i)^2 I)$ *independently.*

Then g_{traj} *and* g_{corr} *are unbiased estimators of* $\nabla_\theta\mathcal{L}_{\text{simple}-\text{reward}}$, *and* $\text{Cov}(g_{\text{corr}}) \preceq \text{Cov}(g_{\text{traj}})$.

Proof. Both estimators are unbiased, since each term $g_{t_i}(X_{t_i}, X_1)$ within each of the summations are identically distributed as $g_t(X_t, X_1)$ for $t \sim U[0,1]$, $(X_t, X_1) \sim p_{t,1}^b$. Since each individual term is unbiased, their average is unbiased

To prove the total ordering of covariances, we make heavy use of the law of total covariance, which lets us express the covariance of two quantities in terms of the covariance of their conditional expectations:

$$\mathrm{Cov}(A, B) = \mathrm{Cov}(\mathbb{E}[A \mid C], \mathbb{E}[B \mid C]) + \mathbb{E}[\mathrm{Cov}(A, B \mid C)]. \tag{46}$$

Note $\mathrm{Cov}(g_{\mathrm{traj}}) = \mathrm{Cov}(\sum_{i=1}^{m} g_{t_i}) = \sum_{i=1}^{m} \mathrm{Cov}(g_{t_i}) + \sum_{i \neq j} \mathrm{Cov}(g_{t_i}, g_{t_j})$, and the same decomposition holds for g_{corr}. The diagonal terms are identical, since each g_{t_i} is distributed exactly the same, hence to compare the covariances it suffices to compare

$$\mathrm{Cov}(g_{t_i}(X_{t_i}, X_1), g_{t_j}(X_{t_j}, X_1)) \qquad \text{versus} \qquad \mathrm{Cov}(g_{t_i}(X'_{t_i}, X_1), g_{t_j}(X'_{t_j}, X_1)). \tag{47}$$

For the first case, when (X_{t_i}, X_{t_j}) are sampled from the same trajectory, then note conditioned on $(X_t)_{t \in [0,1]}$, both g_{t_i} and g_{t_j} are independent, and both have expectation $\int_{s=0}^{1} g_s(X_s, X_1)\mathrm{d}s$.

In the second case, when (X_{t_i}, X_{t_j}) are resampled from the same endpoint X_1, then conditional on X_1, g_{t_i}, g_{t_j} are independent, and they have expectation $\mathbb{E}[\int_{s=0}^{1} g_s(X_s, X_1)\mathrm{d}s \mid X_1]$. Hence, using the law of total covariance multiple times, we obtain

$$\begin{aligned}
\mathrm{Cov}(g_{t_i}(X_{t_i}, X_1), g_{t_j}(X_{t_j}, X_1)) &= \mathrm{Cov}(\textstyle\int_{s=0}^{1} g_s(X_s, X_1)\mathrm{d}s) && (48)\\
&= \mathrm{Cov}(\mathbb{E}[\textstyle\int_{s=0}^{1} g_s(X_s, X_1)\mathrm{d}s \mid X_1]) && (49)\\
&\quad + \mathbb{E}[\mathrm{Cov}(\textstyle\int_{s=0}^{1} g_s(X_s, X_1)\mathrm{d}s \mid X_1)] && (50)\\
&\succeq \mathrm{Cov}(g_{t_i}(X'_{t_i}, X_1), g_{t_j}(X'_{t_j}, X_1)), && (51)
\end{aligned}$$

as the term in eq. (50) is positive semidefinite since it is the expectation of a covariance. □

B Extended Related Work

Non-differentiable rewards *Trajectory* methods cast the generative SDE as a Markov decision process with Gaussian transitions and apply policy gradients to its steps (Black et al., 2024; Fan et al., 2023; Liu et al., 2026; Xue et al., 2025). In contrast, *endpoint and forward-process* methods use complete generations to construct pretraining-like losses. FPO uses an evidence lower bound to estimate the intractable endpoint likelihood (McAllister et al., 2025); other methods derive their losses from stochastic optimal control (Bergmeister et al., 2026) or online reward-weighted regression (Fan et al., 2025).

In text-to-image experiments, RAM reaches a matched reward in up to $50\times$ fewer training steps than Flow-GRPO (Bergmeister et al., 2026). Methods that re-noise generated samples to construct training states are particularly similar to SERF. FDFO also applies an endpoint reward direction to intermediate velocity updates without propagating it through the sampling Jacobian, but estimates that direction from paired generations rather than reward gradients (McAllister et al., 2026).

Differentiable rewards AlignProp and DRaFT optimize differentiable rewards by backpropagating through sampling (Prabhudesai et al., 2023; Clark et al., 2024). DRaFT-K restricts backpropagation to the final K steps, while DRaFT-LV averages gradients over independently noised copies of a generated sample (Clark et al., 2024). ReFL evaluates the reward on a predicted clean image at an intermediate step and differentiates through that prediction (Xu et al., 2023). These methods still differentiate through at least one sampling or prediction step.

VGG-Flow derives guidance from ODE optimal control and trains an auxiliary network to estimate it (Liu et al., 2025a). SQDF approximates soft-Q guidance by evaluating the reward on a clean-image estimate, without training a critic (Kang et al., 2026). Adjoint Matching constructs its training target by integrating adjoint dynamics along sampled trajectories (Domingo-Enrich et al., 2025). SERF also starts from stochastic optimal control but approximates the endpoint reward guidance to obtain a local target without adjoint integration.

Reward Score Matching RSM is particularly close to SERF: both use reward gradients evaluated at fully generated samples to construct local regression targets (Lee et al., 2026). RSM derives its objective from soft reinforcement learning and trains at states collected along the sampling trajectory. SERF instead independently re-noises each generated endpoint to construct training states and derives its target from the stochastic-control solution. Under the conditions in Proposition 2, SERF's target has lower variance than the corresponding RSM target.

Efficient Adjoint Matching EAM shares SERF's use of endpoint noising and its stochastic-control perspective (Shin et al., 2026). It obtains a tractable target by replacing the reference dynamics with a linear base drift and modifying the terminal cost, which requires an estimate of the pretrained model's terminal score. SERF retains the reference dynamics and instead approximates the propagation of the clean-image reward gradient.

C Additional Results

C.1 Quantitative Results

We report a complete trade-off panel in Figure 6.

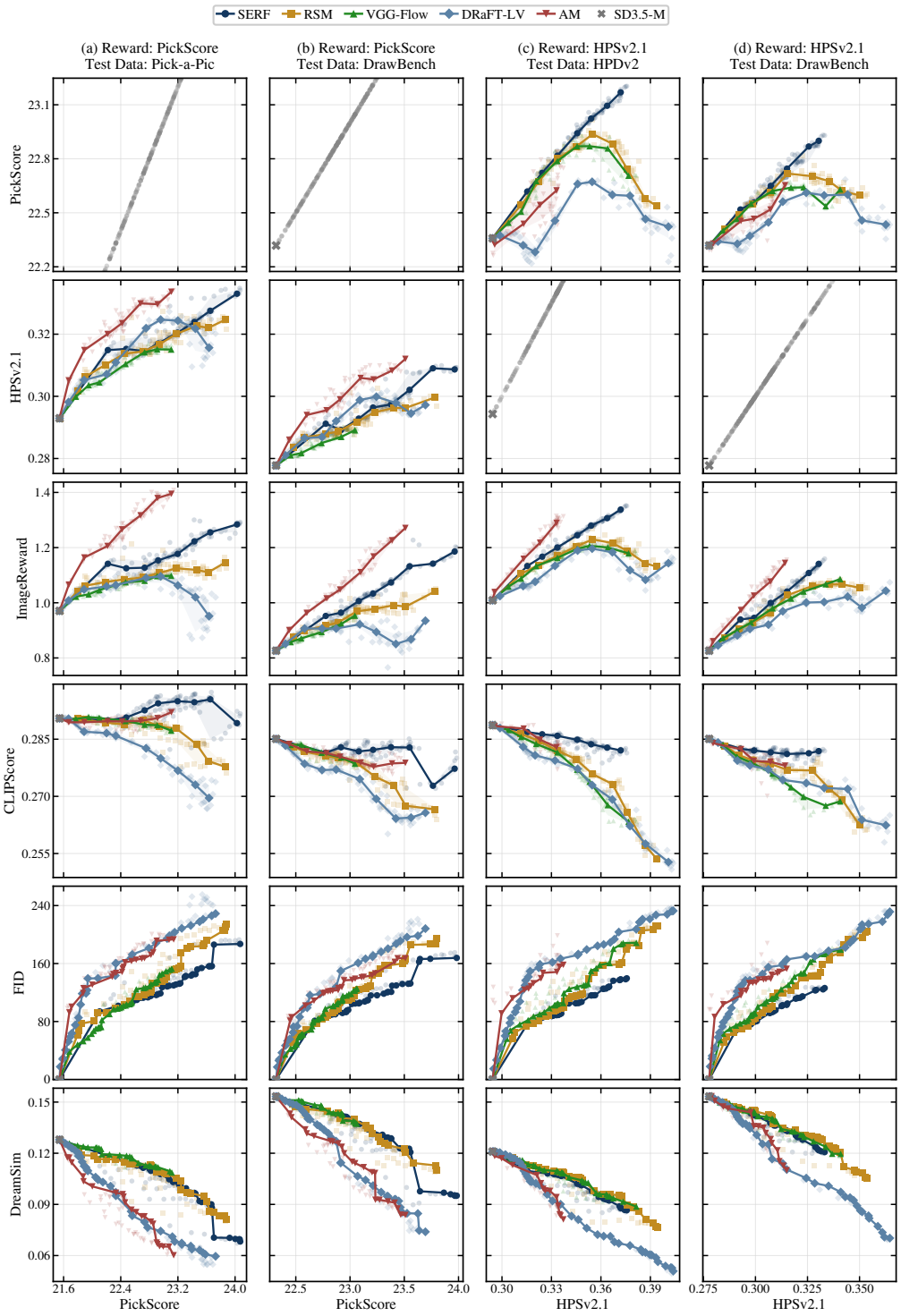

Figure 6: Reward–quality trade-offs.

C.2 Ablations

We report in Figure 7, Figure 8, and Figure 9 training settings ablations for different training rewards and evaluation datasets. Figure 10 examines two choices in constructing the training signal: which noise levels to sample and how to weight the reward-gradient target at each noise level. Low-noise-biased sampling favours states closer to clean data, whereas uniform sampling treats all noise levels equally. Target weighting multiplies the training target by $1-t$, reducing its magnitude near pure noise ($t = 1$) and retaining it near clean data ($t = 0$). We compare three configurations: low-noise-biased sampling without target weighting (our default), low-noise-biased sampling with target weighting, and uniform sampling with target weighting. In the legend, "No" for low-noise sampling therefore denotes uniform sampling, rather than the absence of timestep sampling.

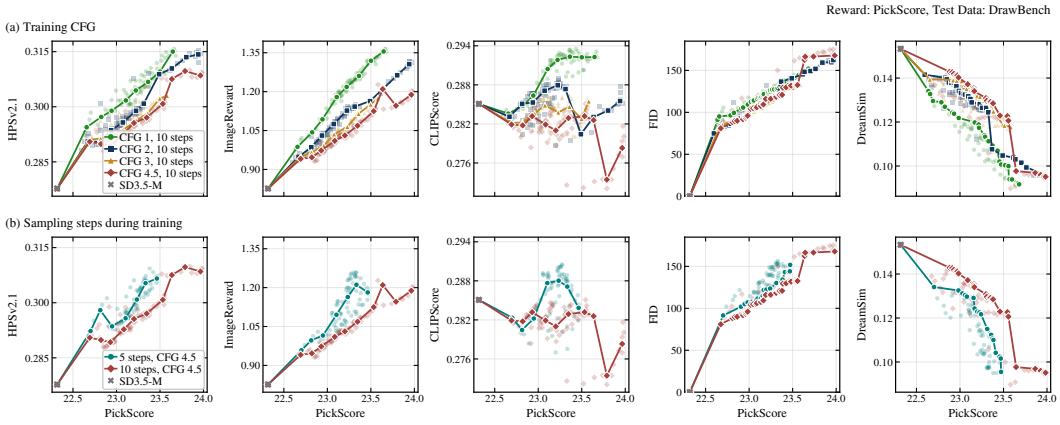

Figure 7: SERF trained on PickScore and evaluated on DrawBench. Top: Comparison for different training CFG scales. Bottom: using five or ten sampling steps per training rollout.

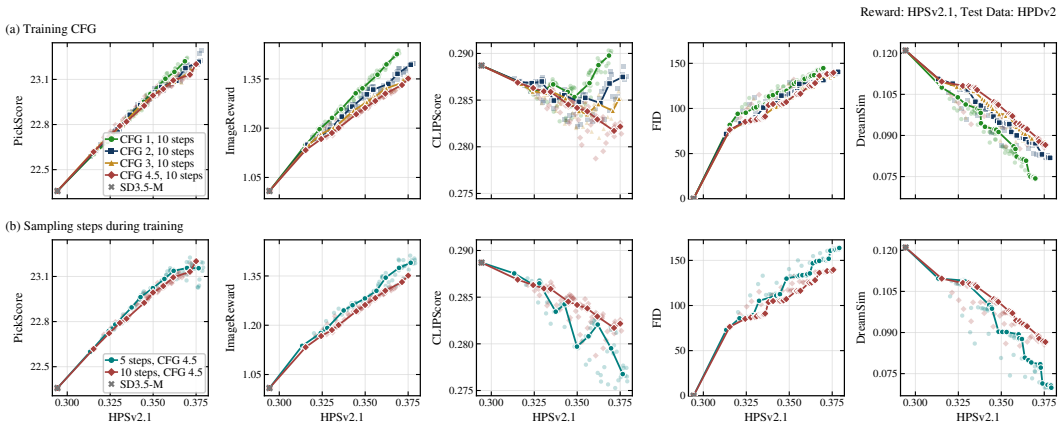

Figure 8: SERF trained on HPSv2.1 and evaluated on HPDv2. Top: Comparison for different training CFG scales. Bottom: using five or ten sampling steps per training rollout.

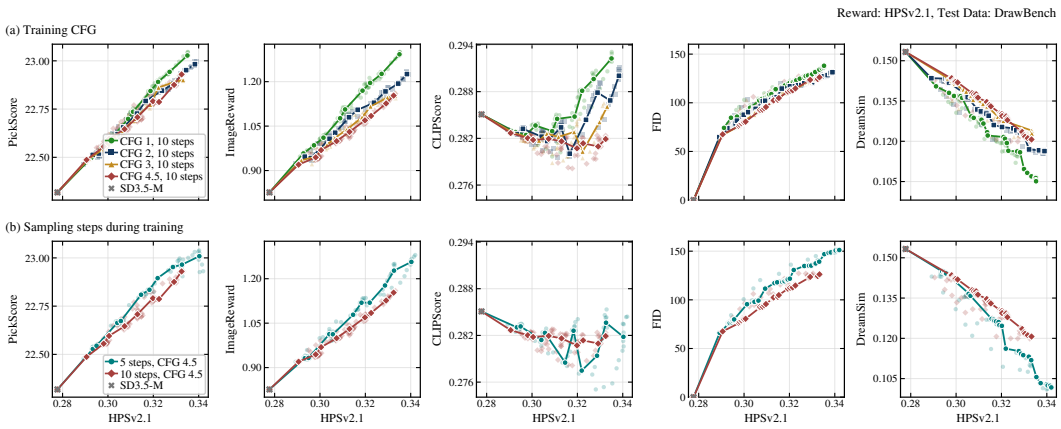

Figure 9: SERF trained on HPSv2.1 and evaluated on DrawBench. Top: Comparison for different training CFG scales. Bottom: using five or ten sampling steps per training rollout.

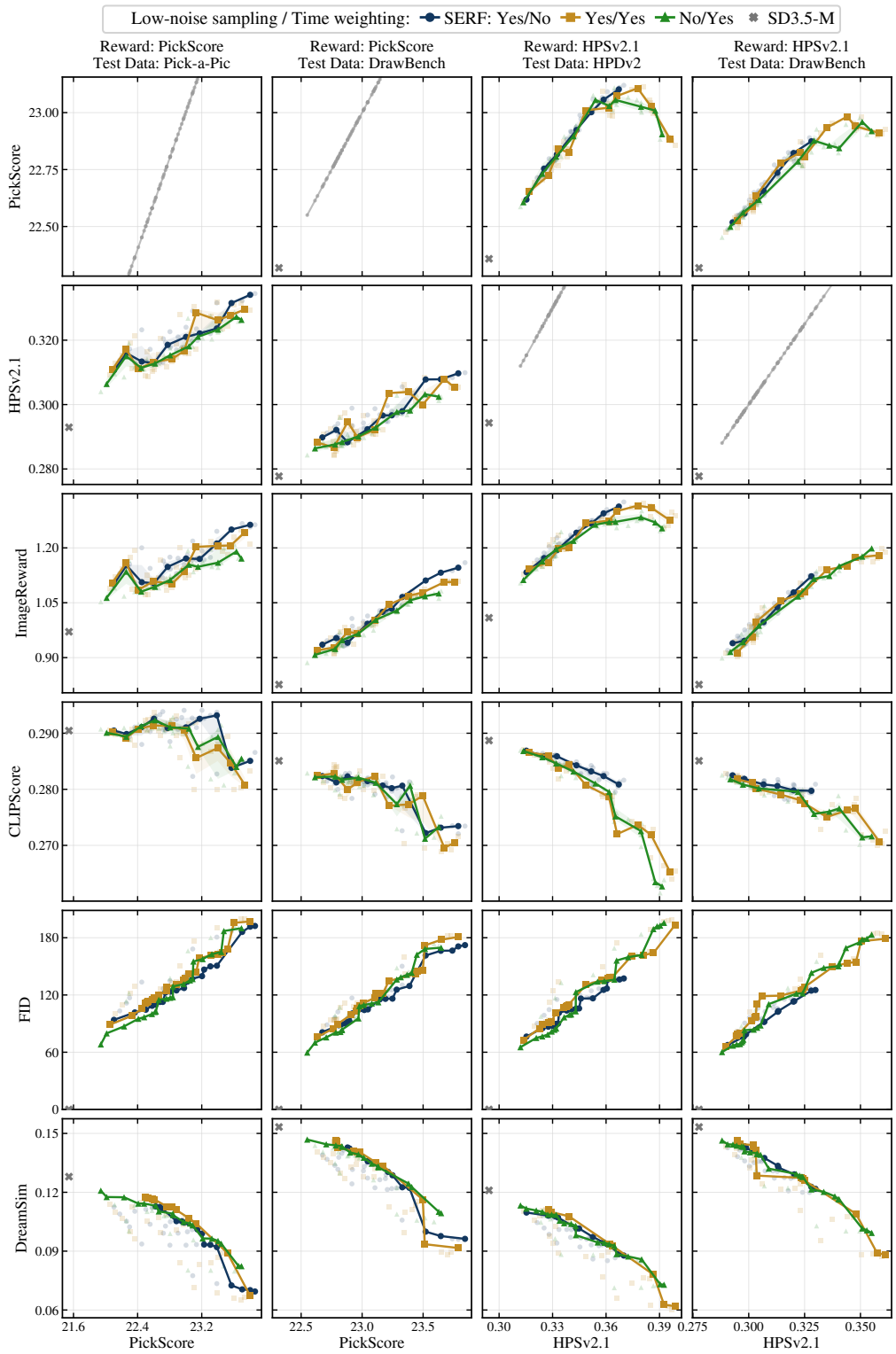

Figure 10: Ablation of timestep sampling and target weighting in SERF. All variants use the common training horizon of 140 updates.

C.3 Qualitative Results

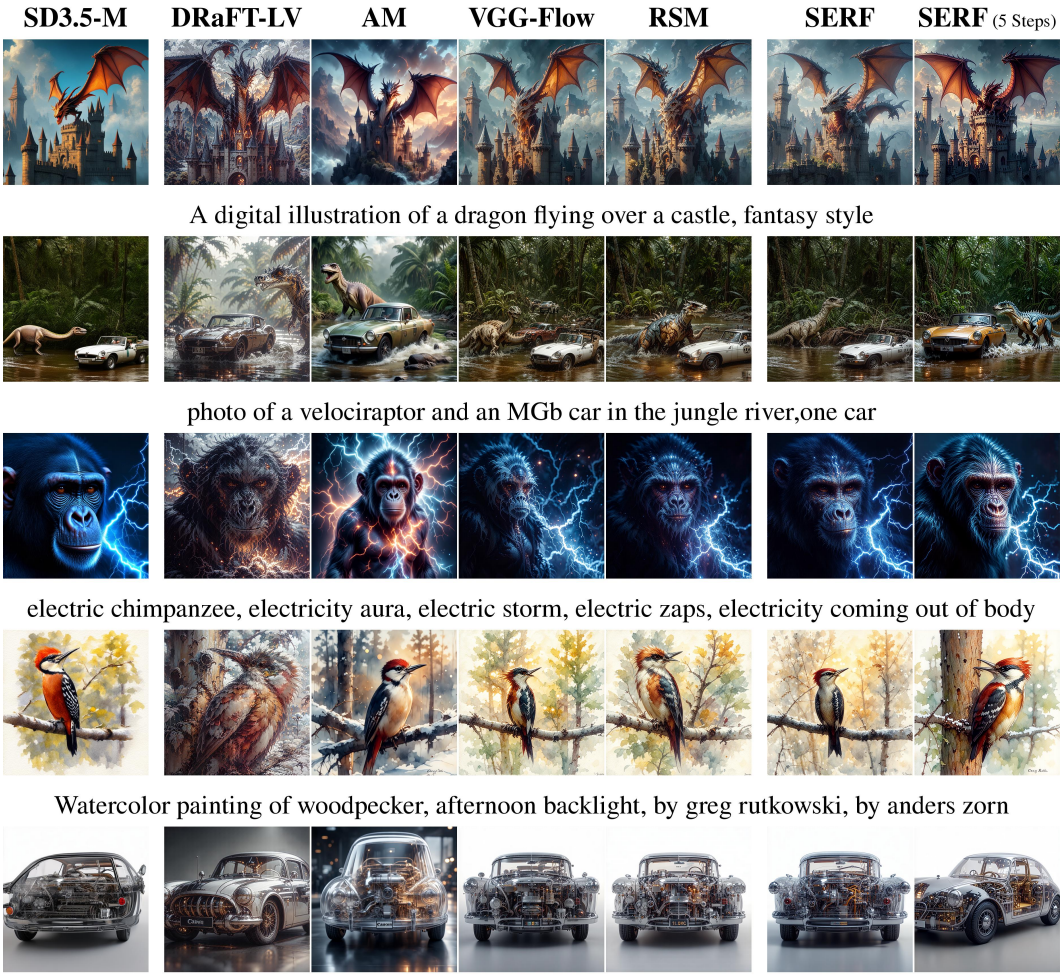

Figure 11: Qualitative comparison of PickScore-trained models on held-out Pick-a-Pic prompts. All methods use the same initial noise, CFG 4.5, and 20 Euler steps at inference. SERF uses 10 training sampling steps unless otherwise indicated.

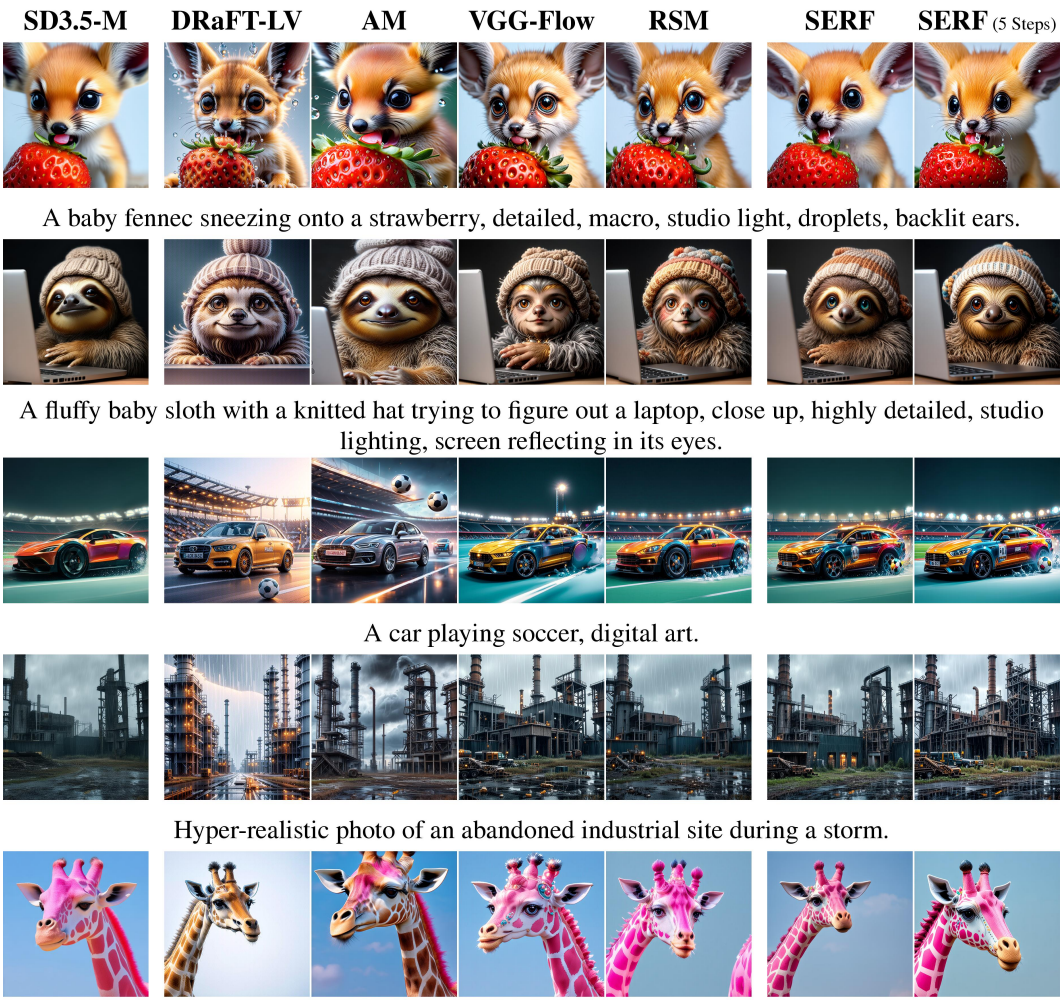

Figure 12: Qualitative comparison of HPSv2.1-trained models on DrawBench prompts. All methods use the same initial noise, CFG 4.5, and 20 Euler steps at inference. SERF uses 10 training sampling steps unless otherwise indicated.

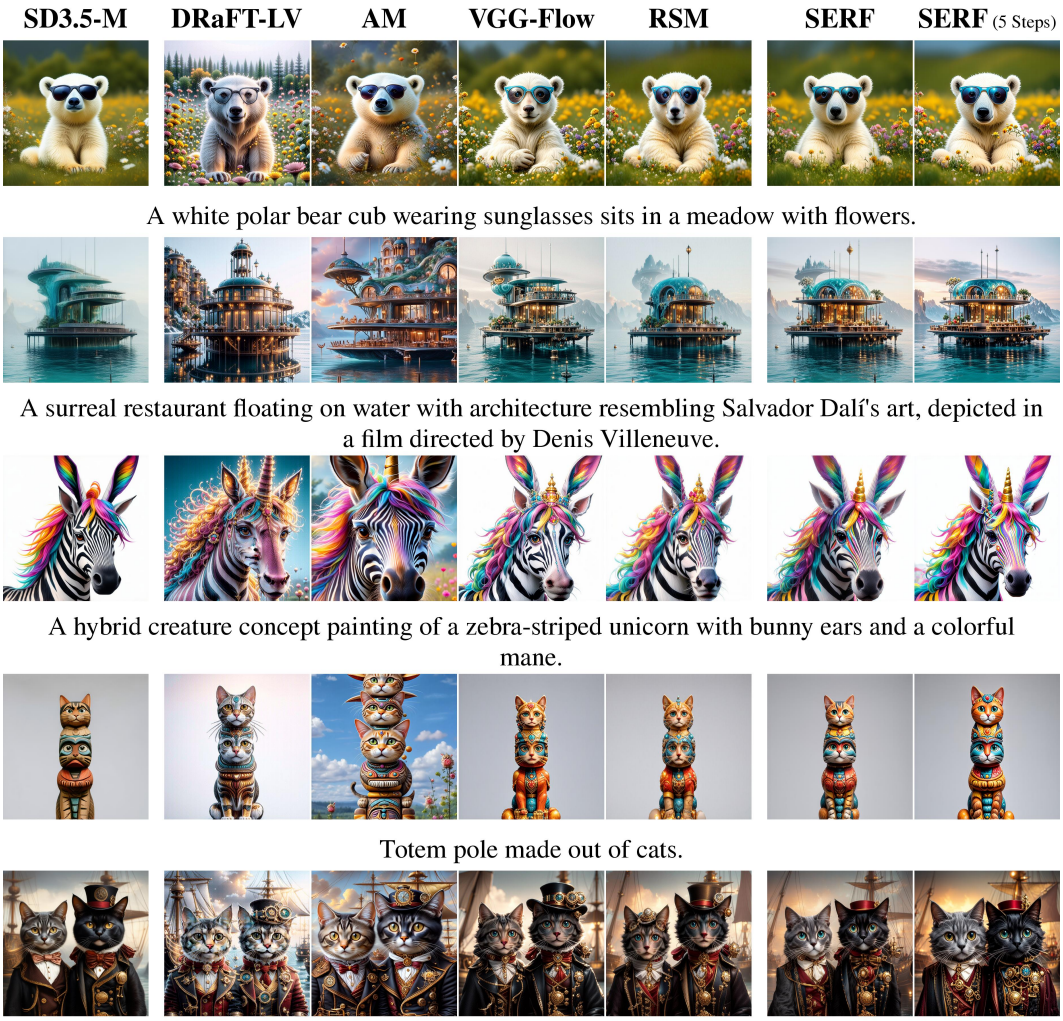

Figure 13: Qualitative comparison of HPSv2.1-trained models on held-out HPDv2 prompts. All methods use the same initial noise, CFG 4.5, and 20 Euler steps at inference. SERF uses 10 training sampling steps unless otherwise indicated.