Flow-DPPO: Divergence Proximal Policy Optimization for Flow Matching Models

Bowen Ping[1,3,4*] **Xiangxin Zhou**[4*†] **Penghui Qi**[5,6]
Minnan Luo[1,2,3‡] **Liefeng Bo**[4] **Tianyu Pang**[4‡]
[1]School of Computer Science and Technology, Xi'an Jiaotong University, China [2]NERC for VI&A
[3]Shaanxi Province Key Lab of BDKE, Xi'an Jiaotong University, Xi'an, 710049, China
[4]Tencent Hunyuan [5]Sea AI Lab [6]National University of Singapore

Abstract

Recent work has demonstrated that online reinforcement learning (RL) can substantially improve the quality and alignment of flow matching models for image and video generation. Methods such as Flow-GRPO and CPS cast the denoising process as a Markov Decision Process and apply PPO-style ratio clipping to enforce a trust region. However, we argue that *ratio clipping is structurally ill-suited for flow models*: the probability ratio between new and old policies is a noisy, single-sample estimate of the true policy divergence, leading to over-constraining in some regions of the trajectory and under-constraining in others. We propose **Flow-DPPO** (Flow Divergence Proximal Policy Optimization), which replaces ratio clipping with a divergence proximal constraint. A key observation is that the per-step policy in flow models is Gaussian, enabling *exact* and *cheap* computation of the KL divergence between old and new policies. Flow-DPPO employs an asymmetric divergence mask that blocks gradient updates only when they simultaneously move away from the trusted region and violate the divergence threshold. Experiments show that Flow-DPPO achieves higher rewards with better KL-proximal efficiency, alleviates catastrophic forgetting, promotes balanced multi-objective optimization, and enables stable multi-epoch training where ratio clipping degrades. Project page is available here.

1 Introduction

Reinforcement learning (RL) has emerged as a core paradigm for aligning models with downstream objectives. In language models, RL methods such as DPO [29] and GRPO [32] have substantially improved alignment [26] and reasoning capabilities [8]. Recently, these advances have been extended to image and video generation [17, 36, 19, 38, 41, 43, 25], including few-step generation [23, 24, 9, 10], where flow matching models [16, 21] represent the dominant generative framework. Among them, Flow-GRPO [17] and DanceGRPO [42] demonstrated strong performance by transforming deterministic ODE sampling into stochastic SDE trajectories and introducing PPO-style ratio clipping to enforce *trust-region optimization*.

The theoretical foundation of trust-region methods originates from Trust Region Policy Optimization (TRPO) [30], which establishes a *policy improvement bound*: monotonic improvement is guaranteed when policy updates remain within a trust region defined by the divergence between the old and new policies. PPO [31] later introduced ratio clipping as a computationally efficient first-order approximation to TRPO. However, as noted by Qi et al. [27], each clipping decision is based on

*Equal Contribution (Work was done during Bowen's internship at Tencent Hunyuan).
†Project Lead. ‡Corresponding author. BDKE: Big Data Knowledge Engineering.
NERC for VI&A: National Engineering Research Center for Visual Information and Applications.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

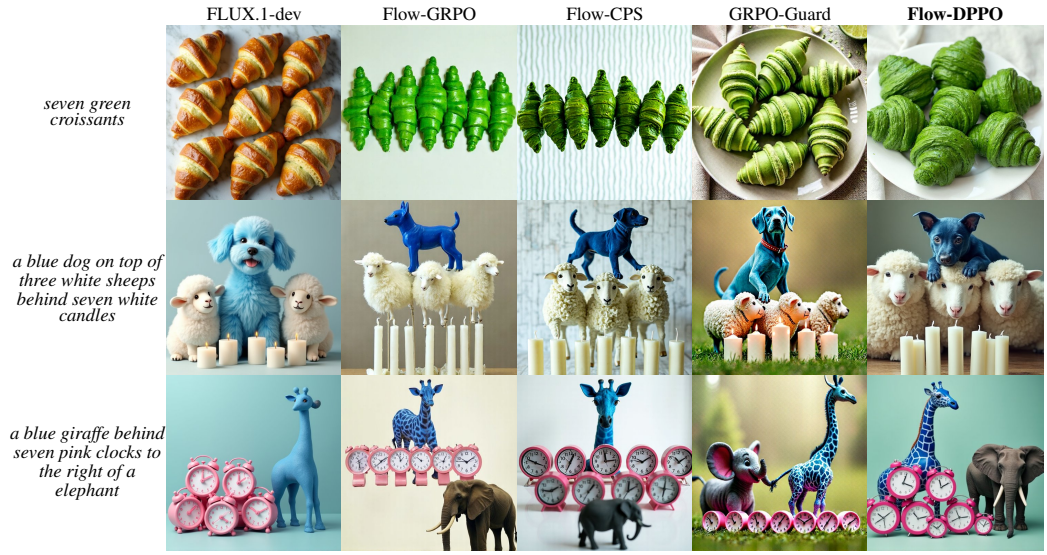


Figure 1: Qualitative comparison on FLUX.1-dev [4] with GenEval2 [12] prompts. Flow-DPPO achieves competitive compositional accuracy with notably less image quality degradation compared to Flow-GRPO [17], Flow-CPS [38], and GRPO-Guard [39], reflecting its superior KL-proximal efficiency.

a single-sample Monte Carlo estimate of the true Total Variation (TV) divergence, rather than the divergence itself. In the continuous and high-dimensional latent space of flow models, this estimation noise becomes substantially amplified and skews the ratio distribution, a shift that GRPO-Guard [39] observed empirically. Although the exact joint ratio satisfies $\mathbb{E}_{\pi_{\theta_{\text{old}}}}[r] = 1$, its log has negative mean, $\mathbb{E}_{\pi_{\theta_{\text{old}}}}[\log r] = -D_{\text{KL}}(\pi_{\theta_{\text{old}}} \| \pi_\theta)$, and off-policy ratios are right-skewed: their median is below one and more than half of their probability mass lies below one. As a result, the symmetric PPO clipping range $[1-\epsilon, 1+\epsilon]$ behaves asymmetrically, failing to adequately constrain over-optimization for positive-advantage samples while excessively clipping negative-advantage ones.

To mitigate this imbalance, GRPO-Guard proposed normalizing the ratio distribution. While this rebalancing alleviates the symptom, it does not address the root cause: the ratio remains a noisy, per-sample proxy for the true policy divergence. We observe that flow models offer a structural advantage that sidesteps this problem entirely. Because each per-step policy is *Gaussian* with a mean $\boldsymbol{\mu}_\theta$ determined by the velocity network and a fixed, schedule-dependent variance σ, the KL divergence between old and new policies reduces to $\|\boldsymbol{\mu}_{\theta_{\text{old}}} - \boldsymbol{\mu}_\theta\|^2/(2\sigma^2)$, which is an *exact*, *deterministic* quantity that can be computed from two forward passes already performed during training. Unlike the LLM setting, where DPPO [27] must resort to approximate divergence reductions over large vocabularies, flow models admit exact divergence computation at no additional cost. This motivates replacing ratio clipping with a direct KL-proximal trust region constraint.

Building on this insight, we propose **Flow-DPPO** (Flow Divergence Proximal Policy Optimization), which replaces ratio clipping with a divergence-based mask. The mask blocks gradient updates only when two conditions are jointly met: (1) the advantage and ratio indicate that the update is moving the policy *away* from the old policy, and (2) the exact KL divergence already exceeds a threshold. This design directly evaluates the per-step divergence at sampled states while preserving the beneficial asymmetric structure of PPO: updates that move the policy *toward* the old policy are never blocked, accelerating recovery from overshooting. Extensive experiments on various base models demonstrate that Flow-DPPO achieves superior reward optimization, improved KL-proximal efficiency, stronger robustness to catastrophic forgetting, balanced multi-objective optimization that mitigates reward hacking, and stable multi-epoch training that enables higher sample efficiency. Figure 1 presents qualitative generation results demonstrating that Flow-DPPO achieves competitive compositional accuracy while preserving notably higher visual quality than existing methods.

2 Preliminaries

Flow matching [16, 21] learns a continuous-time velocity field that transports samples from a simple source distribution to the data distribution. Specifically, let $\boldsymbol{x}_0 \sim \pi_0 = p_{\text{data}}$, and define an

interpolating path $\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, where α_t and σ_t determine the probability path between data and noise. This construction induces a conditional distribution $\pi_{t|0}(\boldsymbol{x}_t \mid \boldsymbol{x}_0) = \mathcal{N}(\alpha_t \boldsymbol{x}_0, \sigma_t^2 \mathbf{I})$. The goal of flow matching is to train a time-dependent vector field $\boldsymbol{v}_\theta(\boldsymbol{x}_t, t)$ to match the target velocity, which is given by $\boldsymbol{v} = \frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \dot{\alpha}_t \boldsymbol{x}_0 + \dot{\sigma}_t \boldsymbol{\epsilon}$, where $\dot{f}_t := \mathrm{d}f_t/\mathrm{d}t$ denotes the time derivative. The model $\boldsymbol{v}_\theta(\boldsymbol{x}_t, t)$ is then trained by minimizing the regression objective

$$\mathbb{E}_{t,\, \boldsymbol{x}_0 \sim \pi_0,\, \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \big[w(t) \| \boldsymbol{v}_\theta(\boldsymbol{x}_t, t) - \boldsymbol{v} \|_2^2 \big], \tag{1}$$

where $w(t)$ is a weighting function. After training, samples are generated by solving the ODE $\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \boldsymbol{v}_\theta(\boldsymbol{x}_t, t)$. In practice, simple numerical solvers such as Euler discretization are often sufficient for high-quality sampling [13, 22, 33]. A notable special case is rectified flow [21], which uses the linear conditional path $\alpha_t = 1 - t$ and $\sigma_t = t$. Under this choice, the target velocity reduces to $\boldsymbol{v} = \boldsymbol{\epsilon} - \boldsymbol{x}_0$. We adopt this linear schedule throughout the paper.

2.1 RL Fine-Tuning for Flow Matching Models

For text-conditional flow matching models, given a conditioning prompt $\boldsymbol{c}$, generation starts from a Gaussian latent $\boldsymbol{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and progressively transforms it into a clean sample $\boldsymbol{x}_0$. At each timestep t, the flow model predicts a velocity field $\boldsymbol{v}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c})$, which specifies a deterministic generation direction. Applying RL algorithms such as GRPO [32] to flow matching models requires a sampler-induced stochastic policy at each denoising step. Flow-GRPO [17] constructs such a policy via an ODE-to-SDE conversion, which transforms the probability-flow ODE into an equivalent SDE with the same marginals [1, 2, 34]: $\mathrm{d}\boldsymbol{x}_t = \Big[\boldsymbol{v}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}) + \frac{\sigma_t^2}{2t} \big(\boldsymbol{x}_t + (1-t) \boldsymbol{v}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}) \big) \Big] \mathrm{d}t + \sigma_t \, \mathrm{d}\boldsymbol{w}$, where $\mathrm{d}\boldsymbol{w}$ denotes Wiener process increments, $\sigma_t = a \sqrt{\frac{t}{1-t}}$, and a is a scalar hyperparameter controlling the noise level. Applying Euler–Maruyama discretization yields the Flow-SDE sampler:

$$\boldsymbol{x}_{t-\Delta t} = \boldsymbol{x}_t + \left[\boldsymbol{v}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}) + \frac{\sigma_t^2}{2t} \big(\boldsymbol{x}_t + (1-t) \boldsymbol{v}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}) \big) \right] \Delta t + \sigma_t \sqrt{\Delta t} \, \boldsymbol{\epsilon}, \tag{2}$$

with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. An alternative is Coefficients-Preserving Sampling (CPS) [38], which reduces the excessive noise injection in Flow-SDE and better preserves the interpolation structure of the scheduler. Let $\hat{\boldsymbol{x}}_0 = \boldsymbol{x}_t - t \, \boldsymbol{v}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}), \hat{\boldsymbol{x}}_1 = \boldsymbol{x}_t + (1-t) \, \boldsymbol{v}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c})$ denote the predicted clean sample and noise component, respectively. CPS updates the latent as

$$\boldsymbol{x}_{t-\Delta t} = (1 - (t - \Delta t)) \, \hat{\boldsymbol{x}}_0 + (t - \Delta t) \cos\Big(\frac{\eta \pi}{2} \Big) \, \hat{\boldsymbol{x}}_1 + (t - \Delta t) \sin\Big(\frac{\eta \pi}{2} \Big) \, \boldsymbol{\epsilon}, \tag{3}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $\eta \in [0, 1]$ controls the stochasticity. Both Flow-SDE and CPS therefore induce Gaussian per-step policies written as

$$p_\theta(\boldsymbol{x}_{t-\Delta t} \mid \boldsymbol{x}_t, t, \boldsymbol{c}) = \mathcal{N}\big(\boldsymbol{x}_{t-\Delta t};\, \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}),\, \sigma^2(t) \mathbf{I} \big), \tag{4}$$

where the specific forms of $\boldsymbol{\mu}_\theta$ and $\sigma(t)$ depend on the sampler. The above generative process can be formulated as a finite-horizon Markov Decision Process (MDP) [3, 7, 17]. To distinguish the discrete decision process from the underlying continuous-time flow, we use $k \in \{1, \ldots, K\}$ for the MDP state index and $t \in [0, 1]$ for the reverse-time variable of the flow. Let $0 = t_K < t_{K-1} < \cdots < t_1 = 1$ be a discretization of reverse time, so that state k corresponds to flow time t_k. The state at step k is $\boldsymbol{s}_k = (\boldsymbol{c}, t_k, \boldsymbol{x}_{t_k})$. Note that $t_k - t_{k+1} = \Delta t$. For $k = 1, \ldots, K-1$, the action is the next latent sample, $\boldsymbol{a}_k = \boldsymbol{x}_{t_{k+1}}$, drawn from the sampler-induced policy $\pi_\theta(\boldsymbol{a}_k \mid \boldsymbol{s}_k) = \pi_\theta(\boldsymbol{x}_{t_{k+1}} \mid \boldsymbol{x}_{t_k}, t_k, \boldsymbol{c})$. Given the sampled action, the transition is deterministic, with next state $\boldsymbol{s}_{k+1} = (\boldsymbol{c}, t_{k+1}, \boldsymbol{x}_{t_{k+1}})$. The rollout starts from $\boldsymbol{c} \sim p(\boldsymbol{c})$ and $\boldsymbol{x}_{t_1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and terminates at $k = K$ where $t_K = 0$.

After the full generative process, a scalar reward $R(\boldsymbol{x}_0, \boldsymbol{c})$ is provided. RL fine-tuning maximizes the expected terminal reward with a KL regularization term that penalizes deviation from the pretrained reference policy π_{ref}: $\max_\theta \; \mathbb{E}_{\boldsymbol{c} \sim p(\boldsymbol{c}),\, \tau \sim \pi_\theta} \Big[R(\boldsymbol{x}_0, \boldsymbol{c}) - \beta \sum_{k=1}^{K-1} D_{\mathrm{KL}}\big(\pi_\theta(\cdot \mid \boldsymbol{s}_k) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid \boldsymbol{s}_k) \big) \Big]$, where $\tau = (\boldsymbol{s}_1, \boldsymbol{a}_1, \boldsymbol{s}_2, \boldsymbol{a}_2, \ldots, \boldsymbol{s}_{K-1}, \boldsymbol{a}_{K-1}, \boldsymbol{s}_K)$ denotes a trajectory induced by π_θ and $\beta \geq 0$ controls the regularization strength. This KL penalty discourages reward hacking and mitigates catastrophic forgetting of the pretrained model's capabilities.

Flow-GRPO [17] applies GRPO to the above MDP. Given a prompt $\boldsymbol{c}$, the current policy generates a group of G samples $\{\boldsymbol{x}_0^i\}_{i=1}^G$. Their rewards are normalized within the group to obtain relative

advantages: $\hat{A}^i = \big(R(\boldsymbol{x}_0^i, \boldsymbol{c}) - \text{mean}(\{R(\boldsymbol{x}_0^j, \boldsymbol{c})\}_{j=1}^G)\big)/\text{std}(\{R(\boldsymbol{x}_0^j, \boldsymbol{c})\}_{j=1}^G)$. In practice, each policy optimization iteration begins by rolling out a batch of data, which is then split into several minibatches for multiple gradient steps. This procedure introduces policy *staleness*: after the first update, the optimizing policy has already diverged from the behavior policy that generated the data. To control this off-policy drift, a trust region mechanism is applied. Following PPO [31], the policy is optimized using the clipped surrogate objective

$$\mathcal{L}^{\text{Flow-GRPO}}(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{K-1}\sum_{k=1}^{K-1}\left(\min\left(r_k^i(\theta)\hat{A}^i,\ \text{clip}(r_k^i(\theta), 1-\epsilon, 1+\epsilon)\hat{A}^i\right)\right)\right], \tag{5}$$

where we omit the KL penalty term $D_{\text{KL}}(\pi_\theta \,\|\, \pi_{\text{ref}})$ for brevity, and the per-step importance ratio is defined as $r_k^i(\theta) = \frac{p_\theta(\boldsymbol{x}^i_{t_k-\Delta t}|\boldsymbol{x}^i_{t_k}, \boldsymbol{c})}{p_{\theta_{\text{old}}}(\boldsymbol{x}^i_{t_k-\Delta t}|\boldsymbol{x}^i_{t_k}, \boldsymbol{c})}$. Since both Flow-SDE and CPS define Gaussian per-step policies as in Eq. (4), the log-ratio admits the same closed-form expression:

$$\log r_k^i(\theta) = \frac{\|\boldsymbol{x}^i_{t_k-\Delta t} - \boldsymbol{\mu}_{\theta_{\text{old}}}\|^2 - \|\boldsymbol{x}^i_{t_k-\Delta t} - \boldsymbol{\mu}_\theta\|^2}{2\sigma^2(t_k)}. \tag{6}$$

Therefore, both samplers can be optimized within the same GRPO framework, differing only in the parameterization of the induced stochastic policy.

3 Methodology

In this section, we first derive a policy improvement bound that justifies trust-region methods for flow models. Then, we show that ratio clipping is a noisy proxy for the true divergence constraint. Finally, we present Flow-DPPO, which leverages exact KL computation to enforce a deterministic, variance-free divergence criterion at sampled states.

3.1 Trust-Region Policy Optimization for Flow Matching Models

Inspired by Schulman et al. [31], Qi et al. [27], we adapt the trust region framework to the flow model fine-tuning setting defined in Section 2.1. This setting differs from the classical discounted RL paradigm in two important ways. First, the problem is an undiscounted episodic task with a finite horizon of $K-1$ decision steps. Second, due to the terminal reward structure, advantages are estimated at the trajectory level rather than per step. These properties necessitate a tailored policy improvement guarantee. We follow the MDP defined in Section 2.1.

Theorem 3.1 (Performance Difference Identity for Flow Models)**.** *In the finite-horizon flow model MDP with $K-1$ decision steps, let $J(\pi) = \mathbb{E}_{\boldsymbol{c}\sim p(\boldsymbol{c}),\,\tau\sim\pi}[R(\boldsymbol{x}_0, \boldsymbol{c})]$ denote the expected reward. For any two policies π_θ and $\pi_{\theta_{\text{old}}}$, the performance difference decomposes as: $J(\pi_\theta) - J(\pi_{\theta_{\text{old}}}) = L'_{\theta_{\text{old}}}(\pi_\theta) - \Delta(\pi_{\theta_{\text{old}}}, \pi_\theta)$, where the surrogate objective is*

$$L'_{\theta_{\text{old}}}(\pi_\theta) = \mathbb{E}_{\tau\sim\pi_{\theta_{\text{old}}}}\left[R(\boldsymbol{x}_0, \boldsymbol{c})\sum_{k=1}^{K-1}\left(\frac{\pi_\theta(\boldsymbol{a}_k \mid \boldsymbol{s}_k)}{\pi_{\theta_{\text{old}}}(\boldsymbol{a}_k \mid \boldsymbol{s}_k)} - 1\right)\right], \tag{7}$$

and the error term is

$$\Delta(\pi_{\theta_{\text{old}}}, \pi_\theta) = \mathbb{E}_{\tau\sim\pi_{\theta_{\text{old}}}}\left[R(\boldsymbol{x}_0, \boldsymbol{c})\sum_{k=1}^{K-1}\left(\frac{\pi_\theta(\boldsymbol{a}_k \mid \boldsymbol{s}_k)}{\pi_{\theta_{\text{old}}}(\boldsymbol{a}_k \mid \boldsymbol{s}_k)} - 1\right)\left(1 - \prod_{j=k+1}^{K-1}\frac{\pi_\theta(\boldsymbol{a}_j \mid \boldsymbol{s}_j)}{\pi_{\theta_{\text{old}}}(\boldsymbol{a}_j \mid \boldsymbol{s}_j)}\right)\right].$$

The surrogate $L'_{\theta_{\text{old}}}(\pi_\theta)$ represents a first-order approximation to the true improvement, while the error term Δ captures higher-order interactions between per-step policy changes. To yield a practical optimization objective, we bound this error term.

Theorem 3.2 (Policy Improvement Bound for Flow Models)**.** *In the finite-horizon flow model MDP with $K-1$ decision steps, the policy improvement is lower-bounded by:*

$$J(\pi_\theta) - J(\pi_{\theta_{\text{old}}}) \geq L'_{\theta_{\text{old}}}(\pi_\theta) - 2\xi(K-1)(K-2)\cdot D_{\text{TV}}^{\max}(\pi_{\theta_{\text{old}}}\|\pi_\theta)^2, \tag{8}$$

where $\xi = \max_{\boldsymbol{x}_0,\boldsymbol{c}}|R(\boldsymbol{x}_0, \boldsymbol{c})|$ is the maximum absolute reward, and $D_{\text{TV}}^{\max}(\pi_{\theta_{\text{old}}}\|\pi_\theta) = \max_{\boldsymbol{s}_k} D_{\text{TV}}\big(\pi_{\theta_{\text{old}}}(\cdot \mid \boldsymbol{s}_k)\|\pi_\theta(\cdot \mid \boldsymbol{s}_k)\big)$ is the maximum per-step Total Variation divergence.

Please refer to Appendix C for the detailed derivation; a tighter bound linear in K is given in Appendix C.3. This bound is structurally analogous to the policy improvement bound for LLMs derived in Qi et al. [27]. It provides a rigorous justification for a max-state trust region in flow model fine-tuning. Our practical mask evaluates divergence only at sampled states, so it is a sampled approximation to this constraint rather than a global monotonic-improvement guarantee; Appendix E.2 states this distinction formally. Similar to TRPO [30], we can solve the following constrained optimization problem to ensure stable learning:

$$\max_{\pi_\theta} \quad L'_{\theta_{\text{old}}}(\pi_\theta), \qquad \text{s.t.} \quad D_{\text{TV}}^{\max}(\pi_{\theta_{\text{old}}} \| \pi_\theta) \leq \delta. \tag{9}$$

Remark 3.3 (Exact Divergence in the Gaussian Setting). For the Gaussian per-step policies in Eq. (4), the TV divergence is a monotone function of the mean displacement:

$$D_{\text{TV}}\big(\pi_{\theta_{\text{old}}}(\cdot \mid \boldsymbol{s}_k) \| \pi_\theta(\cdot \mid \boldsymbol{s}_k)\big) = 2\Phi\left(\frac{\|\boldsymbol{\mu}_{\theta_{\text{old}}} - \boldsymbol{\mu}_\theta\|}{2\sigma(t_k)}\right) - 1, \tag{10}$$

where Φ is the standard normal CDF. Constraining the TV divergence below a threshold is therefore equivalent to constraining $\|\boldsymbol{\mu}_{\theta_{\text{old}}} - \boldsymbol{\mu}_\theta\|^2 \leq \delta'$ for an appropriate δ', which is precisely the divergence measure that Flow-DPPO employs. Moreover, the Pinsker inequality $D_{\text{TV}}(p\|q)^2 \leq \frac{1}{2} D_{\text{KL}}(p\|q)$ ensures that a uniform per-step KL constraint also upper-bounds the maximum TV divergence: if $D_{\text{KL}} \leq \delta$ at every state, then $D_{\text{TV}}^{\max} \leq \sqrt{\delta/2}$. In the Gaussian equal-covariance case, the converse also holds since KL and TV are both monotone functions of $\|\boldsymbol{\mu}_{\theta_{\text{old}}} - \boldsymbol{\mu}_\theta\|/\sigma$. The implemented mask checks this relation at sampled states; it does not establish the uniform premise by itself. Unlike the LLM setting, where the discrete vocabulary requires approximate divergence computations [27], the Gaussian structure of flow models provides *exact* per-step divergence at zero additional cost.

3.2 Pitfalls of Ratio Clipping in Flow-GRPO

Flow-GRPO adopts PPO-style ratio clipping to enforce a trust region. For consistency with the Flow-GRPO notation [17], in this and the following subsections we index denoising steps by the flow time t (equivalently, $t = t_k$ in the MDP indexing of Section 2.1). The clipping condition $|r_t^i - 1| \leq \epsilon$ is intended to prevent the new policy from deviating too far from the old one. However, the probability ratio is a fundamentally noisy proxy for the true policy divergence. By definition of the Total Variation divergence,

$$D_{\text{TV}}\big(\pi_{\theta_{\text{old}}}(\cdot \mid \boldsymbol{x}_t) \,\|\, \pi_\theta(\cdot \mid \boldsymbol{x}_t)\big) = \frac{1}{2}\, \mathbb{E}_{\boldsymbol{x}_{t-\Delta t} \sim \pi_{\theta_{\text{old}}}}\big[\, |r_t^i - 1| \,\big], \tag{11}$$

so each individual $|r_t^i - 1|$ is merely a *single-sample Monte Carlo estimate* of $2\, D_{\text{TV}}$. While the policy improvement bound (Theorem 3.2) calls for constraining $D_{\text{TV}}^{\max}$, ratio clipping constrains this noisy per-sample surrogate instead. This issue was identified by Qi et al. [27] in the LLM setting; we now show that the resulting pathology is particularly severe in flow models due to the high-dimensional continuous action space.

Recall from Eq. (6) that the log-ratio is:

$$\log r_t^i(\theta) = \frac{\|\boldsymbol{x}_{t-\Delta t}^i - \boldsymbol{\mu}_{\theta_{\text{old}}}\|^2 - \|\boldsymbol{x}_{t-\Delta t}^i - \boldsymbol{\mu}_\theta\|^2}{2\sigma^2}. \tag{12}$$

Since $\boldsymbol{x}_{t-\Delta t}^i$ is sampled from $\mathcal{N}(\boldsymbol{\mu}_{\theta_{\text{old}}}, \sigma^2 \mathbf{I})$, we can write $\boldsymbol{x}_{t-\Delta t}^i = \boldsymbol{\mu}_{\theta_{\text{old}}} + \sigma\boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Substituting and letting $\boldsymbol{d} = \boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\theta_{\text{old}}}$:

$$\log r_t^i(\theta) = \frac{\|\sigma\boldsymbol{\epsilon}\|^2 - \|\sigma\boldsymbol{\epsilon} - \boldsymbol{d}\|^2}{2\sigma^2} = \frac{2\sigma\, \boldsymbol{\epsilon}^\top \boldsymbol{d} - \|\boldsymbol{d}\|^2}{2\sigma^2} = \frac{\boldsymbol{\epsilon}^\top \boldsymbol{d}}{\sigma} - \frac{\|\boldsymbol{d}\|^2}{2\sigma^2}. \tag{13}$$

The first term, $\boldsymbol{\epsilon}^\top \boldsymbol{d}/\sigma$, is a zero-mean random variable with variance $\|\boldsymbol{d}\|^2/\sigma^2$. This reveals that the log-ratio is dominated by *noise*: the signal (the deterministic second term $-\|\boldsymbol{d}\|^2/(2\sigma^2)$) is exactly the negative of the KL divergence, but it is corrupted by a noise term whose standard deviation $\|\boldsymbol{d}\|/\sigma$ is of the same order as the signal itself. This analysis yields two key insights:

1. **High variance.** The ratio r_t^i is inherently noisy due to the stochastic sample $\boldsymbol{\epsilon}$. Even when the true KL divergence $\|\boldsymbol{d}\|^2/(2\sigma^2)$ is moderate, individual ratio samples can be extreme (either very large or very small), triggering spurious clipping.

2. **Noise-dependent clipping.** Whether an update is clipped depends heavily on the random noise $\boldsymbol{\epsilon}$ drawn during sampling, rather than the true policy divergence. Two trajectories with identical policy parameters but different noise realizations may receive entirely different clipping decisions.

In contrast, the true KL divergence $D_{\mathrm{KL}}(\pi_{\theta_{\text{old}}}\|\pi_\theta) = \|\boldsymbol{d}\|^2/(2\sigma^2)$ is a *deterministic* function of the policy parameters alone, unaffected by the sampling noise. This motivates our approach: replace the noisy ratio-based trust region with a direct divergence constraint. A detailed variance analysis is provided in Appendix E.

3.3 Divergence Proximal Policy Optimization for Flow Models

We now derive the divergence between old and new policies in the flow model setting and present our **Flow-DPPO** algorithm.

Exact KL divergence. Since both $\pi_{\theta_{\text{old}}}(\cdot \mid \boldsymbol{x}_t)$ and $\pi_\theta(\cdot \mid \boldsymbol{x}_t)$ are Gaussians with the same variance $\sigma^2\mathbf{I}$ but different means, the KL divergence admits the closed form (see Appendix D for derivation):

$$D_{\mathrm{KL}}\big(\pi_{\theta_{\text{old}}}(\cdot \mid \boldsymbol{x}_t) \,\|\, \pi_\theta(\cdot \mid \boldsymbol{x}_t)\big) = \frac{\|\boldsymbol{\mu}_{\theta_{\text{old}}}(\boldsymbol{x}_t, t) - \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t)\|^2}{2\sigma^2}. \tag{14}$$

In implementation, both log probability and Gaussian KL are averaged over the n latent dimensions: $D_t = D_{\mathrm{KL}}(\pi_{\theta_{\text{old}}}\|\pi_\theta)/n$. Consequently, δ is a per-dimension threshold. For Flow-SDE (corresponding to Eq. (2)), $\sigma^2 = \sigma_t^2 \Delta t$, giving:

$$D_{\mathrm{KL}}^{\text{SDE}}(\pi_{\theta_{\text{old}}}\|\pi_\theta) = \frac{\Delta t}{2}\left(\frac{\sigma_t(1-t)}{2t} + \frac{1}{\sigma_t}\right)^2 \|\boldsymbol{v}_\theta(\boldsymbol{x}_t, t) - \boldsymbol{v}_{\theta_{\text{old}}}(\boldsymbol{x}_t, t)\|^2. \tag{15}$$

For CPS (corresponding to Eq. (3)), with $\sigma_{\text{CPS}} = (t - \Delta t)\sin(\eta\pi/2)$:

$$D_{\mathrm{KL}}^{\text{CPS}}(\pi_{\theta_{\text{old}}}\|\pi_\theta) = \frac{\|\boldsymbol{\mu}_\theta^{\text{CPS}}(\boldsymbol{x}_t, t) - \boldsymbol{\mu}_{\theta_{\text{old}}}^{\text{CPS}}(\boldsymbol{x}_t, t)\|^2}{2(t - \Delta t)^2 \sin^2(\eta\pi/2)}. \tag{16}$$

Remark 3.4. In the LLM setting, DPPO [27] must approximate the true divergence via Binary or Top-K reductions of the vocabulary distribution, as computing exact TV or KL over $|\mathcal{V}| > 100\text{K}$ tokens is memory-prohibitive. In flow models, the Gaussian policy structure yields *exact* divergence at negligible cost, namely the squared difference between two forward passes of the velocity network. This makes divergence-based trust regions strictly more natural for flow models than for LLMs.

The Flow-DPPO mask. We define the Flow-DPPO objective as:

$$\mathcal{L}^{\text{Flow-DPPO}}(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{K-1}\sum_{k=1}^{K-1}\Big(M_{t_k}^i \cdot r_{t_k}^i(\theta)\cdot \hat{A}^i - \beta D_{\mathrm{KL}}(\pi_\theta\|\pi_{\text{ref}})\Big)\right], \tag{17}$$

where the divergence-based mask is:

$$M_t^i = \begin{cases} 0, & \text{if } \big(\hat{A}^i > 0 \text{ and } r_t^i > 1 \text{ and } D_t > \delta\big) \\ & \quad \text{or } \big(\hat{A}^i < 0 \text{ and } r_t^i < 1 \text{ and } D_t > \delta\big), \\ 1, & \text{otherwise,} \end{cases} \tag{18}$$

with $D_t \equiv D_{\mathrm{KL}}\big(\pi_{\theta_{\text{old}}}(\cdot \mid \boldsymbol{x}_t^i) \,\|\, \pi_\theta(\cdot \mid \boldsymbol{x}_t^i)\big)/n$ and δ a per-dimension divergence threshold.

Asymmetric design. The mask in Eq. (18) preserves the asymmetric structure that makes PPO effective. It only blocks updates that are *already moving away* from the old policy:

- When $\hat{A}^i > 0$ and $r_t^i > 1$: the gradient is pushing the policy *further* from θ_{old} (increasing an already-increased action probability). The mask blocks this if the divergence exceeds δ.
- When $\hat{A}^i < 0$ and $r_t^i < 1$: the gradient is decreasing an already-decreased action probability, again moving *away* from the old policy. The mask blocks this if divergence exceeds δ.
- In all other cases ($\hat{A}^i > 0, r_t^i < 1$ or $\hat{A}^i < 0, r_t^i > 1$): the gradient is moving the policy *toward* the old policy. These beneficial updates are *never* blocked, regardless of the divergence level.

This asymmetry ensures that the trust region constraint does not impede recovery: when the policy has drifted too far, corrective updates remain uninhibited. We provide a justification of this directional condition and discuss refined mask variants in Appendix F.

Table 1: Performance comparison on SD3.5 and FLUX2-9B. The training is applied on the In-Domain (GenEval2). The Out-of-Domain (PickScore) prompts are only used for evaluation. The corresponding training curves are in Figures 2 and 8. The full version including single-reward training is in Table 4.

	In-Domain (GenEval2)				Out-of-Domain (PickScore)		
Model	GenEval2	CLIP	PickScore	HPSv2	CLIP	PickScore	HPSv2
Pretrained baselines (before RL)							
SD3.5-medium	12.4	0.250	21.00	0.213	0.244	19.99	0.210
FLUX2-klein-base-9B	25.4	0.281	20.92	0.228	0.254	20.05	0.230
FLUX.1-dev	23.3	0.297	23.26	0.315	0.276	21.91	0.304
SD3.5-medium, multi-reward RL fine-tuning							
Flow-GRPO	39.9	0.358	25.09	0.399	0.273	22.07	0.349
Flow-CPS	44.6	0.359	25.51	0.407	0.265	22.08	0.343
GRPO-Guard	47.8	0.353	25.64	0.409	0.272	22.32	0.354
Diffusion-NFT	42.5	0.334	25.30	0.394	0.269	22.52	0.355
Flow-DPPO	48.1	0.345	25.63	0.409	0.273	**22.58**	0.360
Flow-DPPO + CPS	**51.6**	**0.369**	**25.72**	**0.415**	**0.279**	22.51	**0.361**
FLUX2-klein-base-9B, multi-reward RL fine-tuning							
Flow-GRPO	46.8	0.371	25.61	0.412	0.277	22.62	0.357
Flow-CPS	47.1	0.361	25.70	0.416	0.276	22.85	0.364
GRPO-Guard	49.0	0.375	25.27	0.411	0.269	21.99	0.349
Diffusion-NFT	47.3	0.336	24.87	0.389	0.274	22.47	0.351
Flow-DPPO	**57.7**	0.364	25.76	0.418	0.282	22.90	0.368
Flow-DPPO + CPS	55.2	**0.386**	**26.15**	**0.427**	**0.287**	**22.97**	**0.370**

4 Experiments

Models and Baselines. We employ Stable Diffusion 3.5 Medium [6, 35] (SD3.5), FLUX2-klein-base-9B [5] (FLUX2-9B) and FLUX.1-dev [4] as base models to cover diverse architectures and scales. We compare our method against four competitive baselines: Flow-GRPO [17], Flow-CPS [38], GRPO-Guard [39] and Diffusion-NFT [43]. Specifically, we evaluate two variants of our approach: Flow-DPPO (using SDE sampling from Flow-GRPO) and Flow-DPPO+CPS (using CPS-scheduled SDE sampling). Detailed configurations, including the complete seed, batch, checkpoint, and KL-reduction conventions, are deferred to Appendix G.

Metrics and Datasets. GenEval2 [12] and PickScore [14] are selected as in-domain and out-of-domain (OOD) datasets, respectively. For GenEval2, we follow the official template to generate 20k synthetic training prompts and evaluate on the 800 officially released prompts. To monitor catastrophic forgetting under distribution shifts, we track PickScore [14], CLIP [28] score, and HPSv2 [40] during training. We report results for both single-reward optimization (GenEval2 only) and multi-reward training, where GDPO [20] aggregates advantages with equal reward weights.

4.1 Main results

Performance and Generalization. As summarized in Table 1, Flow-DPPO variants consistently outperform all baselines across both base models and all evaluation metrics, with particularly substantial gains in the GenEval2 reward. In the single-reward setting (optimizing GenEval2 only), Figure 2 demonstrates that our proposed variants not only achieve superior performance on FLUX2-9B compared to baselines but also exhibit a more stable training trajectory. These empirical advantages persist across SD3.5 (Figure 7) and FLUX.1-dev (Figure 9).

We attribute this superiority to the precise divergence-based mask in Flow-DPPO. By mitigating the influence of samples falling outside the trust region, which are susceptible to reward hacking, Flow-DPPO maintains a more robust optimization gradient. This constraint prevents the model from excessively exploiting individual rewards at the expense of others, thereby achieving a superior balance across multiple optimization objectives and fostering stable convergence. This is further corroborated by the multi-reward training curves in Figure 2, where Flow-DPPO variants consistently outperform all baselines across most metrics on SD3.5, without sacrificing any individual objective.

Extension to Video Generation. We further evaluate Flow-DPPO on Wan2.1-T2V-1.3B [37] with VideoReward [18]. The Flow-GRPO and Flow-DPPO runs both use CPS sampling with $\eta = 0.9$ and a matched budget of 1024 videos per epoch and differ only in ratio clipping versus the asymmetric KL mask. At epochs 100 and 200, Flow-DPPO obtains higher VideoReward with 21–35% lower reference KL, and its VideoReward remains higher at epoch 300. This matched

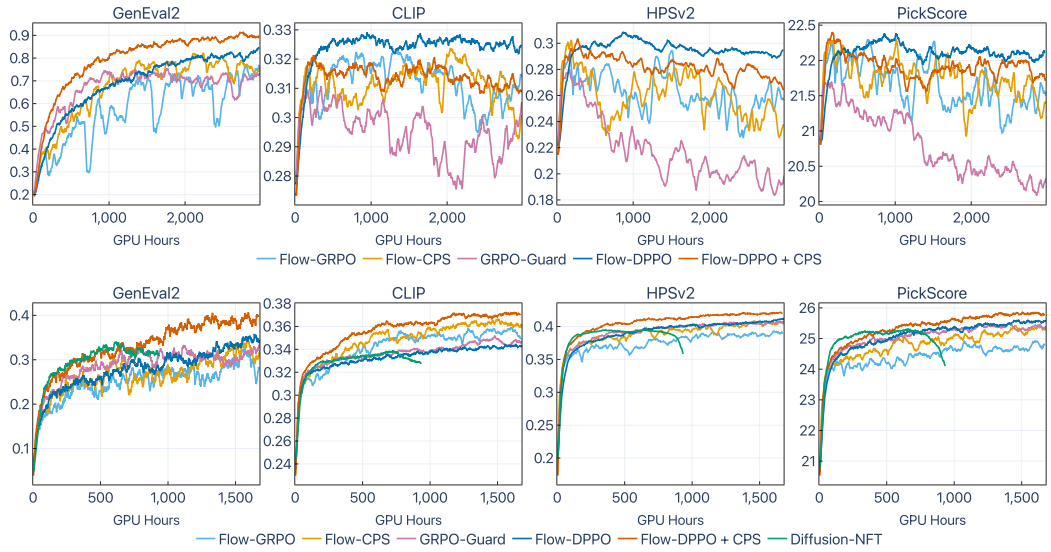


Figure 2: Top: training curves on FLUX2-9B with single-reward optimization. **Bottom:** training curves on SD3.5 with multi-reward optimization, where Diffusion-NFT [43] is included as an additional baseline. Flow-DPPO variants consistently achieve stronger reward optimization with less out-of-domain forgetting across both settings.

experiment reproduces the image-domain reward–KL trend in video generation; complete protocols are provided in Appendix G.3.

Out-of-domain Behavior and Catastrophic Forgetting. To investigate catastrophic forgetting, we analyze OOD metrics (PickScore, CLIP, and HPSv2) and the KL divergence from the pre-trained model. As illustrated in Figure 2, OOD metrics initially increase across all methods as RL optimization drives the model toward higher visual quality. However, as training progresses, these metrics decline, indicating that the model overfits the in-domain reward (GenEval2) at the expense of OOD knowledge. Notably, Flow-DPPO variants exhibit significantly less OOD degradation, suggesting that catastrophic forgetting is effectively mitigated. Qualitative results in Figure 6 further support this, demonstrating that our methods better preserve visual fidelity on OOD prompts. Consistently, Table 2 shows that Flow-DPPO variants maintain a lower KL divergence in most settings. This reduced distribution drift aligns with OOD metric trends, collectively indicating stronger resistance to reward hacking and forgetting. Ultimately, these results highlight that the divergence-based mask acts as a safety boundary, allowing the model to learn from rewards without losing its original generative quality or falling into distribution collapse.

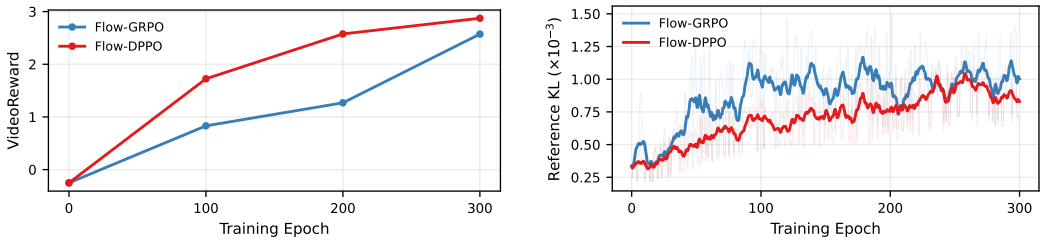
The endpoint divergence provides a complementary distribution-level check: Flow-DPPO has lower reference KL than its matched ratio-clipping baseline in nine of ten sampler–setting comparisons. The exception is FLUX2-9B single-reward training with CPS, where Flow-CPS has lower final reference KL.

Table 2: KL divergence ($\times 10^{-3}$) between the RL fine-tuned model and the pre-trained reference. Lower is better. Full curves are in Figure 10.

	FLUX2-9B			SD3.5	
Method	Single	Multi	+CFG	Single	Multi
Flow-SDE schedule					
Flow-GRPO	0.77	0.79	1.36	2.34	3.81
GRPO-Guard	1.07	1.01	1.63	2.05	3.33
Flow-DPPO	**0.17**	**0.49**	**0.51**	**1.16**	**2.49**
CPS schedule					
Flow-CPS	**0.24**	1.66	1.51	2.41	3.18
Flow-DPPO + CPS	0.68	**0.70**	**0.83**	**1.60**	**2.52**

4.2 Analysis

Asymmetric Masking and Divergence Threshold. At each sampled state, we partition updates into three groups: (1) in-region updates with $D \leq \delta$; (2) out-of-region updates with $D > \delta$ whose direction pulls the sample back toward the trust region; and (3) out-of-region updates that push it farther away. Flow-DPPO retains groups (1) and (2) and masks group (3). Figure 4 analyzes this



(a) Evaluation VideoReward. **(b)** Reference KL during training.

Figure 3: Matched Wan2.1-T2V-1.3B video-generation experiment over the shared 300-epoch horizon. VideoReward is evaluated every 100 epochs. Reference KL is measured from the pretrained model.

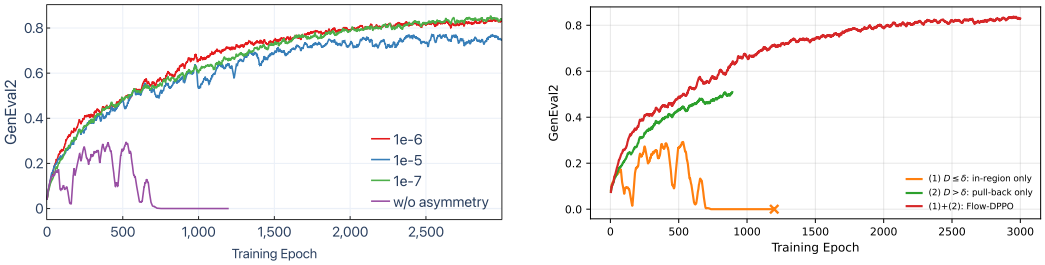


Figure 4: Trust-region ablations on SD3.5 with CPS sampling and single-reward GenEval2. **Left:** threshold sensitivity and the effect of removing asymmetry. **Right:** decomposition into in-region updates (group 1), out-of-region pull-back updates (group 2), and their combination in Flow-DPPO. The in-region-only run collapses, while pull-back-only updates remain stable but make less progress than the complete mask.

design on SD3.5 with CPS sampling. Without asymmetry, only group (1) is retained and training collapses because no retained update can pull a drifting policy back. Keeping only group (2) remains stable over the observed horizon but underperforms the complete mask, showing that pull-back updates provide stability while in-region updates supply reward progress. Their combination yields stable long-term improvement.

Regarding the divergence threshold, a looser threshold (10^{-5}) results in diminished stability and suboptimal convergence. A tighter threshold (10^{-7}) initially slows learning but yields greater stability and slightly better final performance.

Multi-epoch Training and Sample Efficiency. Given the high computational cost of rollouts, we investigate how sample reuse frequency affects optimization efficiency on SD3.5. We vary (i) the number of groups per rollout and (ii) the number of training epochs per rollout (inner loops), which determines sample reuse frequency. For instance, two inner loops reuse each rollout batch for two consecutive gradient steps.

While our main experiments use 64 groups with 1 inner loop (G64-I1), we further explore two efficiency-oriented settings: G32-I2 (half the rollout computation with samples reused twice) and G64-I2 (standard rollout computation with doubled training intensity). As shown in Figure 5, baseline methods (Flow-GRPO, Flow-CPS) struggle to achieve sustained gains under multi-epoch training, often leading to performance plateaus or degradation. In contrast, Flow-DPPO variants successfully reuse rollout samples across multiple updates, yielding consistent long-term performance improvements. This advantage stems from the divergence-based mask, which constrains updates within the trust region, ensuring efficient sample utilization. This offers a promising direction for scenarios where rollouts are computationally expensive, such as long-video generation.

Additional Validation. Appendix H further reports training- and evaluation-seed variation and PickScore-only optimization (Appendix H.3), multi-sample divergence estimation (Appendix H.4), and direct Gaussian-KL penalty and oracle early-stopping baselines (Appendix H.5). In all of these checks, the main findings still hold; full settings, tables, and curves are given in the appendix.

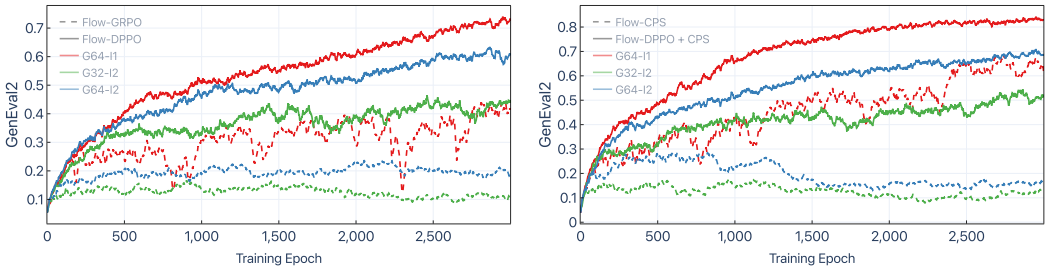


Figure 5: Multi-epoch training on SD3.5 (Left: Flow-SDE, Right: CPS). Flow-DPPO variants show consistent long-term gains under multi-epoch training (G64-I2 and G32-I2), while baselines plateau or even degrade.

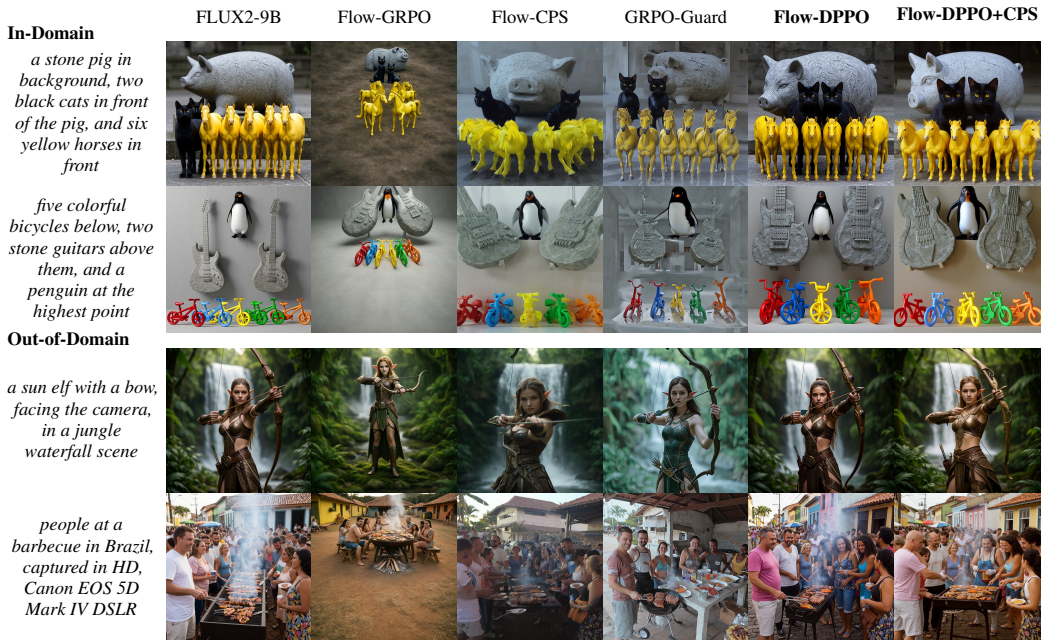


Figure 6: Qualitative comparison on FLUX2-9B with single-reward setting and controlled seeds for each prompt at the same training iteration. Flow-DPPO and Flow-DPPO + CPS retain competitive in-domain performance with less reward hacking while exhibiting notably less catastrophic forgetting on out-of-domain prompts.

5 Conclusion

We show ratio clipping in flow models is a noisy, single-sample proxy for divergence. To address this, we propose a divergence-based mask using the *exact* KL at *zero extra cost*. Across multiple base models, sampling schedules, and reward objectives, Flow-DPPO consistently achieves superior performance to baselines in terms of reward optimization and catastrophic forgetting. Furthermore, Flow-DPPO enables stable multi-epoch training where ratio clipping degrades, and the additional video experiment supports its broader applicability beyond image generation.

Acknowledgment

This work is supported by the Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (No. JYB2025XDXM101), the National Natural Science Foundation of China (No. 62272374, No. 62192781), the Natural Science Foundation of Shaanxi Province (No. 2024JC-JCQN-62), the State Key Laboratory of Communication Content Cognition under Grant No. A202502, and the Key Research and Development Project in Shaanxi Province (No. 2023GXLH-024).

Competing Interests. The authors declare no competing interests.

References

[1] Michael Samuel Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*, 2023. 3

[2] Michael Samuel Albergo, Mark Goldstein, Nicholas Matthew Boffi, Rajesh Ranganath, and Eric Vanden-Eijnden. Stochastic interpolants with data-dependent couplings. In *International Conference on Machine Learning*, pages 921–937. PMLR, 2024. 3

[3] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024. 3

[4] Black Forest Labs. Flux.1: Announcing black forest labs. https://blackforestlabs.ai/announcing-black-forest-labs/, 2024. 2, 7

[5] Black Forest Labs. FLUX.2 [klein]: Towards interactive visual intelligence. https://bfl.ai/blog/flux2-klein-towards-interactive-visual-intelligence, 2026. Model weights: https://huggingface.co/black-forest-labs/FLUX.2-klein-base-9B. 7

[6] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024. 7

[7] Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Reinforcement learning for fine-tuning text-to-image diffusion models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 3

[8] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. 1

[9] Yushi Huang, Xiangxin Zhou, Ruoyu Wang, Chi Zhang, Jun Zhang, and Tianyu Pang. Reinforcing few-step generators via reward-tilted distribution matching. *arXiv preprint arXiv:2605.26108*, 2026. 1

[10] Yushi Huang, Xiangxin Zhou, Jun Zhang, Liefeng Bo, and Tianyu Pang. MeanFlowNFT: Bringing forward-process RL to average-velocity generators. *arXiv preprint arXiv:2607.15273*, 2026. 1

[11] Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the nineteenth international conference on machine learning*, pages 267–274, 2002. 15

[12] Amita Kamath, Kai-Wei Chang, Ranjay Krishna, Luke Zettlemoyer, Yushi Hu, and Marjan Ghazvininejad. Geneval 2: Addressing benchmark drift in text-to-image evaluation. *arXiv preprint arXiv:2512.16853*, 2025. 2, 7, 26

[13] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, 2022. 3

[14] 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. *Advances in neural information processing systems*, 36:36652–36663, 2023. 7

[15] Junzhe Li, Yutao Cui, Tao Huang, Yinping Ma, Chun Fan, Yiming Cheng, Miles Yang, Zhao Zhong, and Liefeng Bo. Mixgrpo: Unlocking flow-based grpo efficiency with mixed ode-sde. *arXiv preprint arXiv:2507.21802*, 2025. 24

[16] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. 1, 2

[17] 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. *arXiv preprint arXiv:2505.05470*, 2025. 1, 2, 3, 5, 7

[18] Jie Liu, Gongye Liu, Jiajun Liang, Ziyang Yuan, Xiaokun Liu, Mingwu Zheng, Xiele Wu, Qiulin Wang, Wenyu Qin, Menghan Xia, Xintao Wang, Xiaohong Liu, Fei Yang, Pengfei Wan, Di Zhang, Kun Gai, Yujiu Yang, and Wanli Ouyang. Improving video generation with human feedback. *arXiv preprint arXiv:2501.13918*, 2025. 7

[19] Luping Liu, Chao Du, Tianyu Pang, Zehan Wang, Chongxuan Li, and Dong Xu. Improving long-text alignment for text-to-image diffusion models. In *International Conference on Learning Representations*, 2025. 1

[20] Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, et al. Gdpo: Group reward-decoupled normalization policy optimization for multi-reward rl optimization. *arXiv preprint arXiv:2601.05242*, 2026. 7

[21] Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. 1, 2, 3

[22] 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. In *Advances in Neural Information Processing Systems*, 2022. 3

[23] Yihong Luo, Tianyang Hu, Jiacheng Sun, Yujun Cai, and Jing Tang. Learning few-step diffusion models by trajectory distribution matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17719–17728, 2025. 1

[24] Yihong Luo, Tianyang Hu, Weijian Luo, and Jing Tang. TDM-R1: Reinforcing few-step diffusion models with non-differentiable reward. In *International Conference on Machine Learning*, 2026. 1

[25] Yihong Luo, Tianyang Hu, and Jing Tang. Reinforcing diffusion models by direct group preference optimization. In *International Conference on Learning Representations*, 2026. 1

[26] 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

[27] Penghui Qi, Xiangxin Zhou, Zichen Liu, Tianyu Pang, Chao Du, Min Lin, and Wee Sun Lee. Rethinking the trust region in llm reinforcement learning. *arXiv preprint arXiv:2602.04879*, 2026. 1, 2, 4, 5, 6, 15, 21

[28] 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. 7

[29] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023. 1

[30] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015. 1, 5, 15

[31] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 1, 4

[32] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. 1, 3

[33] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. 3

[34] 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. 3

[35] Stability AI. Stable diffusion 3.5. https://stability.ai/news/introducing-stable-diffusion-3-5, 2024. Model weights: https://huggingface.co/stabilityai/stable-diffusion-3.5-medium. 7

[36] Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8228–8238, 2024. 1

[37] Wan Team, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Li, Pandeng Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 7

[38] Feng Wang and Zihao Yu. Coefficients-preserving sampling for reinforcement learning with flow matching. *arXiv preprint arXiv:2509.05952*, 2025. 1, 2, 3, 7, 15, 20

[39] Jing Wang, Jiajun Liang, Jie Liu, Henglin Liu, Gongye Liu, Jun Zheng, Wanyuan Pang, Ao Ma, Zhenyu Xie, Xintao Wang, et al. Grpo-guard: Mitigating implicit over-optimization in flow matching via regulated clipping. *arXiv preprint arXiv:2510.22319*, 2025. 2, 7, 21

[40] 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. 7

[41] Shuchen Xue, Chongjian Ge, Shilong Zhang, Yichen Li, and Zhi-Ming Ma. Advantage weighted matching: Aligning rl with pretraining in diffusion models. *arXiv preprint arXiv:2509.25050*, 2025. 1

[42] 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. 1

[43] 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 *The Fourteenth International Conference on Learning Representations*, 2026. 1, 7, 8, 25, 28

Appendix of Flow-DPPO: Divergence Proximal Policy Optimization for Flow Matching Models

A Broader Impacts **15**

B The Flow-DPPO Algorithm **15**

C Policy Improvement Bound for Flow Models **15**

C.1 Proof of Performance Difference Identity 15

C.2 Proof of Policy Improvement Bound 16

C.3 A Tighter Policy Improvement Bound 18

C.4 Connection to Gaussian Per-Step Divergence 18

D KL Divergence Between Gaussian Policies **19**

D.1 General Gaussian KL Divergence 19

D.2 Connection Between KL and TV in the Gaussian Setting 19

D.3 Application to Flow-SDE 19

D.4 Application to CPS 19

E Ratio Distribution and Scope of the Guarantee **20**

E.1 Exact and Dimension-Averaged Ratios 20

E.2 Group Normalization and Sampled-State Masking 20

E.3 Relation to Prior Divergence Masks and Broader Samplers 21

F Towards a Predictive Divergence Mask **21**

F.1 Predicting Post-Update Divergence 22

F.2 The Predictive Mask 22

F.3 Discussion on Mask Variants 23

G Experimental Details **23**

G.1 Computational Resources. 23

G.2 Hyperparameters. 23

G.3 Training and Evaluation Protocol 24

H Additional Experimental Results **24**

H.1 Full Results on Image Generation 24

H.1.1 Quantitative Summary on GenEval2 24

H.1.2 Additional Training Curves 24

H.2 KL Divergence Curves 24
H.3 Robustness of the Main Results 28
H.3.1 Variation Across Training and Evaluation Seeds 28
H.3.2 A Second Preference Reward 28
H.3.3 Classifier-Free Guidance 28
H.4 Analysis of the Divergence Mask 29
H.4.1 Which Updates Stabilize Training? 29
H.4.2 Multi-Sample Divergence Estimation 30
H.4.3 Generality Beyond the Timestep-Weighted KL 30
H.5 Alternative Forms of KL Control 30
H.5.1 Exact Gaussian-KL Penalty 30
H.5.2 Oracle KL Early Stopping 31
H.5.3 Reference KL Regularization Strength 31

A Broader Impacts

This work proposes a training algorithm for aligning flow matching models with human preferences via reinforcement learning. On the positive side, improved alignment methods can enhance the quality, controllability, and safety of image generation systems. On the negative side, advances in generative model fine-tuning could be misused to produce misleading or harmful visual content such as deepfakes. We note that Flow-DPPO is a foundational algorithmic contribution and does not introduce new generative capabilities beyond those already present in the base models it fine-tunes; the risks are therefore inherited from, and no greater than, those of the underlying pre-trained models.

B The Flow-DPPO Algorithm

We summarize the complete Flow-DPPO training procedure. The algorithm supports either Flow-SDE or CPS sampling [38] for trajectory generation, uses group-relative advantage estimation, and applies the divergence-based mask during policy optimization.

Computational overhead. The current-policy forward pass that produces $\boldsymbol{\mu}_\theta(\boldsymbol{x}_t^i, t)$ is already required to compute the log ratio. Given the stored old-policy mean, computing the sampler-specific D_t from Eq. 15 or 16 adds only elementwise arithmetic and no additional network forward pass.

C Policy Improvement Bound for Flow Models

We adapt the classical policy improvement theory [11, 30] to the finite-horizon, undiscounted setting of flow model denoising, following the approach of Qi et al. [27] for the LLM regime. We use the MDP notation introduced in Section 2.1: $K-1$ decision steps indexed by $k \in \{1, \dots, K-1\}$, states $\boldsymbol{s}_k = (\boldsymbol{c}, t_k, \boldsymbol{x}_{t_k})$, actions $\boldsymbol{a}_k = \boldsymbol{x}_{t_{k+1}}$, and terminal reward $R(\boldsymbol{x}_0, \boldsymbol{c})$.

C.1 Proof of Performance Difference Identity

Proof of Theorem 3.1. We begin by expressing the performance difference via its definition. Since the reward is only a function of the terminal state $\boldsymbol{x}_0$ and the prompt $\boldsymbol{c}$, we have:

$$\begin{aligned} J(\pi_\theta) - J(\pi_{\theta_{\text{old}}}) &= \mathbb{E}_{\tau\sim\pi_\theta}[R(\boldsymbol{x}_0, \boldsymbol{c})] - \mathbb{E}_{\tau\sim\pi_{\theta_{\text{old}}}}[R(\boldsymbol{x}_0, \boldsymbol{c})] \\ &= \int \big(\pi_\theta(\tau \mid \boldsymbol{c}) - \pi_{\theta_{\text{old}}}(\tau \mid \boldsymbol{c})\big) R(\boldsymbol{x}_0, \boldsymbol{c})\, \mathrm{d}\tau, \end{aligned}$$

where the integral is over all trajectories $\tau = (\boldsymbol{a}_1, \dots, \boldsymbol{a}_{K-1})$ (we omit the deterministic transition structure for notational clarity).

Algorithm 1 Flow-DPPO Training

Input: Flow model $\boldsymbol{v}_\theta$, reference model $\boldsymbol{v}_{\text{ref}}$, reward function R, prompts $\mathcal{C}$
Hyperparameters: group size G, divergence threshold δ, KL coefficient β, stochasticity η
$\theta_{\text{old}} \leftarrow \theta$
for each training iteration **do**
  Sample prompts $\{\boldsymbol{c}_j\} \sim \mathcal{C}$
  // Rollout phase (with θ_{old})
  for each prompt $\boldsymbol{c}_j$ **do**
    Generate G trajectories $\{(\boldsymbol{x}_T^i, \ldots, \boldsymbol{x}_0^i)\}_{i=1}^G$ via Flow-SDE or CPS (Eqs. 2, 3) using $\boldsymbol{v}_{\theta_{\text{old}}}$
    Record log-probabilities $\log p_{\theta_{\text{old}}}(\boldsymbol{x}_{t-\Delta t}^i \mid \boldsymbol{x}_t^i)$ and means $\boldsymbol{\mu}_{\theta_{\text{old}}}(\boldsymbol{x}_t^i, t)$
    Compute rewards $R(\boldsymbol{x}_0^i, \boldsymbol{c}_j)$ and advantages $\hat{A}^i$
  end for
  // Policy optimization phase
  for each gradient step **do**
    Compute current means $\boldsymbol{\mu}_\theta(\boldsymbol{x}_t^i, t)$ via forward pass of $\boldsymbol{v}_\theta$
    Compute $D_t = D_{\text{KL}}(\pi_{\theta_{\text{old}}}(\cdot \mid \boldsymbol{x}_t^i) \| \pi_\theta(\cdot \mid \boldsymbol{x}_t^i))/n$ via Eq. 15 or 16
    Compute ratio $r_t^i(\theta)$ from log-probabilities
    Compute mask M_t^i (Eq. 18)
    Update θ by maximizing $\mathcal{L}^{\text{Flow-DPPO}}$ (Eq. 17)
  end for
  $\theta_{\text{old}} \leftarrow \theta$
end for

The core of the proof is the telescoping identity for the difference in trajectory probabilities. Since $\pi_\theta(\tau \mid \boldsymbol{c}) = \prod_{k=1}^{K-1} \pi_\theta(\boldsymbol{a}_k \mid \boldsymbol{s}_k)$, we apply the algebraic identity $\prod_{k=1}^N a_k - \prod_{k=1}^N b_k = \sum_{k=1}^N \left(\prod_{j=1}^{k-1} b_j\right)(a_k - b_k)\left(\prod_{j=k+1}^N a_j\right)$:

$$\pi_\theta(\tau \mid \boldsymbol{c}) - \pi_{\theta_{\text{old}}}(\tau \mid \boldsymbol{c}) = \sum_{k=1}^{K-1} \left(\prod_{j=1}^{k-1} \pi_{\theta_{\text{old}}}(\boldsymbol{a}_j \mid \boldsymbol{s}_j)\right) \cdot \left(\pi_\theta(\boldsymbol{a}_k \mid \boldsymbol{s}_k) - \pi_{\theta_{\text{old}}}(\boldsymbol{a}_k \mid \boldsymbol{s}_k)\right) \left(\prod_{j=k+1}^{K-1} \pi_\theta(\boldsymbol{a}_j \mid \boldsymbol{s}_j)\right).$$

Substituting into the performance difference and converting to an expectation under $\pi_{\theta_{\text{old}}}$:

$$J(\pi_\theta) - J(\pi_{\theta_{\text{old}}}) = \mathbb{E}_{\tau \sim \pi_{\theta_{\text{old}}}} \left[R(\boldsymbol{x}_0, \boldsymbol{c}) \sum_{k=1}^{K-1} \left(\frac{\pi_\theta(\boldsymbol{a}_k \mid \boldsymbol{s}_k)}{\pi_{\theta_{\text{old}}}(\boldsymbol{a}_k \mid \boldsymbol{s}_k)} - 1 \right) \left(\prod_{j=k+1}^{K-1} \frac{\pi_\theta(\boldsymbol{a}_j \mid \boldsymbol{s}_j)}{\pi_{\theta_{\text{old}}}(\boldsymbol{a}_j \mid \boldsymbol{s}_j)} \right) \right].$$

We decompose this expression by adding and subtracting the term where the future ratio product is set to 1:

$$J(\pi_\theta) - J(\pi_{\theta_{\text{old}}}) = \underbrace{\mathbb{E}_{\tau \sim \pi_{\theta_{\text{old}}}} \left[R(\boldsymbol{x}_0, \boldsymbol{c}) \sum_{k=1}^{K-1} \left(\frac{\pi_\theta(\boldsymbol{a}_k \mid \boldsymbol{s}_k)}{\pi_{\theta_{\text{old}}}(\boldsymbol{a}_k \mid \boldsymbol{s}_k)} - 1 \right) \right]}_{L'_{\theta_{\text{old}}}(\pi_\theta)}$$
$$- \underbrace{\mathbb{E}_{\tau \sim \pi_{\theta_{\text{old}}}} \left[R(\boldsymbol{x}_0, \boldsymbol{c}) \sum_{k=1}^{K-1} \left(\frac{\pi_\theta(\boldsymbol{a}_k \mid \boldsymbol{s}_k)}{\pi_{\theta_{\text{old}}}(\boldsymbol{a}_k \mid \boldsymbol{s}_k)} - 1 \right) \left(1 - \prod_{j=k+1}^{K-1} \frac{\pi_\theta(\boldsymbol{a}_j \mid \boldsymbol{s}_j)}{\pi_{\theta_{\text{old}}}(\boldsymbol{a}_j \mid \boldsymbol{s}_j)} \right) \right]}_{\Delta(\pi_{\theta_{\text{old}}}, \pi_\theta)}.$$

This completes the proof. □

C.2 Proof of Policy Improvement Bound

Lemma C.1 (Bound on Trajectory-Level TV Divergence)**.** *Let $\pi_{\theta_{\text{old}}}$ and π_θ be two policies for the flow model MDP. Let $\pi_{\theta_{\text{old}},>k}(\cdot \mid \boldsymbol{s}_{k+1})$ and $\pi_{\theta,>k}(\cdot \mid \boldsymbol{s}_{k+1})$ denote the distributions over future*

sub-trajectories $(\boldsymbol{a}_{k+1}, \ldots, \boldsymbol{a}_{K-1})$ *starting from state* $\boldsymbol{s}_{k+1}$*. Then:*

$$D_{\mathrm{TV}}\big(\pi_{\theta_{\mathrm{old}},>k}(\cdot \mid \boldsymbol{s}_{k+1}) \| \pi_{\theta,>k}(\cdot \mid \boldsymbol{s}_{k+1})\big) \leq \sum_{j=k+1}^{K-1} \mathbb{E}_{\boldsymbol{s}_j \sim \pi_{\theta_{\mathrm{old}}}} \left[D_{\mathrm{TV}}\big(\pi_{\theta_{\mathrm{old}}}(\cdot \mid \boldsymbol{s}_j) \| \pi_\theta(\cdot \mid \boldsymbol{s}_j)\big)\right],$$

where the expectation is over states visited under $\pi_{\theta_{\mathrm{old}}}$ *starting from* $\boldsymbol{s}_{k+1}$*.*

Proof. Let $P(\tau_{>k}) = \pi_{\theta_{\mathrm{old}},>k}(\tau_{>k} \mid \boldsymbol{s}_{k+1})$ and $Q(\tau_{>k}) = \pi_{\theta,>k}(\tau_{>k} \mid \boldsymbol{s}_{k+1})$, where $\tau_{>k} = (\boldsymbol{a}_{k+1}, \ldots, \boldsymbol{a}_{K-1})$. We have:

$$2D_{\mathrm{TV}}(P\|Q) = \int |P(\tau_{>k}) - Q(\tau_{>k})| \, \mathrm{d}\tau_{>k} = \int \left| \prod_{j=k+1}^{K-1} \pi_{\theta_{\mathrm{old}}}(\boldsymbol{a}_j \mid \boldsymbol{s}_j) - \prod_{j=k+1}^{K-1} \pi_\theta(\boldsymbol{a}_j \mid \boldsymbol{s}_j) \right| \mathrm{d}\tau_{>k}.$$

Applying the telescoping identity $|a_1 \cdots a_N - b_1 \cdots b_N| \leq \sum_{j=1}^{N} \left(\prod_{i=1}^{j-1} a_i\right) |a_j - b_j| \left(\prod_{i=j+1}^{N} b_i\right)$ (which follows from the triangle inequality) and integrating:

$$2D_{\mathrm{TV}}(P\|Q) \leq \sum_{j=k+1}^{K-1} \int \left(\prod_{i=k+1}^{j-1} \pi_{\theta_{\mathrm{old}}}(\boldsymbol{a}_i \mid \boldsymbol{s}_i) \right) |\pi_{\theta_{\mathrm{old}}}(\boldsymbol{a}_j \mid \boldsymbol{s}_j) - \pi_\theta(\boldsymbol{a}_j \mid \boldsymbol{s}_j)| \left(\prod_{i=j+1}^{K-1} \pi_\theta(\boldsymbol{a}_i \mid \boldsymbol{s}_i) \right) \mathrm{d}\tau_{>k}.$$

For each term indexed by j, integrating out the future actions $\boldsymbol{a}_{j+1}, \ldots, \boldsymbol{a}_{K-1}$ yields 1 (since π_θ is normalized), leaving:

$$2D_{\mathrm{TV}}(P\|Q) \leq \sum_{j=k+1}^{K-1} \int \left(\prod_{i=k+1}^{j-1} \pi_{\theta_{\mathrm{old}}}(\boldsymbol{a}_i \mid \boldsymbol{s}_i) \right) \left(\int |\pi_{\theta_{\mathrm{old}}}(\boldsymbol{a}_j \mid \boldsymbol{s}_j) - \pi_\theta(\boldsymbol{a}_j \mid \boldsymbol{s}_j)| \, \mathrm{d}\boldsymbol{a}_j \right) \mathrm{d}\boldsymbol{a}_{k+1} \cdots \mathrm{d}\boldsymbol{a}_{j-1}.$$

The inner integral is $2D_{\mathrm{TV}}(\pi_{\theta_{\mathrm{old}}}(\cdot \mid \boldsymbol{s}_j) \| \pi_\theta(\cdot \mid \boldsymbol{s}_j))$, and the outer integral defines an expectation over states $\boldsymbol{s}_j$ under policy $\pi_{\theta_{\mathrm{old}}}$. Thus:

$$D_{\mathrm{TV}}(P\|Q) \leq \sum_{j=k+1}^{K-1} \mathbb{E}_{\boldsymbol{s}_j \sim \pi_{\theta_{\mathrm{old}}}} \left[D_{\mathrm{TV}}\big(\pi_{\theta_{\mathrm{old}}}(\cdot \mid \boldsymbol{s}_j) \| \pi_\theta(\cdot \mid \boldsymbol{s}_j)\big)\right]. \qquad \square$$

Proof of Theorem 3.2. From Theorem 3.1, we start with the exact performance difference identity:

$$J(\pi_\theta) - J(\pi_{\theta_{\mathrm{old}}}) = L'_{\theta_{\mathrm{old}}}(\pi_\theta) - \Delta(\pi_{\theta_{\mathrm{old}}}, \pi_\theta).$$

Our goal is to upper-bound $|\Delta(\pi_{\theta_{\mathrm{old}}}, \pi_\theta)|$. We begin by bounding the reward by its maximum absolute value $\xi = \max_{\boldsymbol{x}_0, \boldsymbol{c}} |R(\boldsymbol{x}_0, \boldsymbol{c})|$:

$$\begin{aligned} &|\Delta(\pi_{\theta_{\mathrm{old}}}, \pi_\theta)| \\ &\leq \xi \cdot \mathbb{E}_{\tau \sim \pi_{\theta_{\mathrm{old}}}} \left[\sum_{k=1}^{K-1} \left| \frac{\pi_\theta(\boldsymbol{a}_k \mid \boldsymbol{s}_k)}{\pi_{\theta_{\mathrm{old}}}(\boldsymbol{a}_k \mid \boldsymbol{s}_k)} - 1 \right| \cdot \left| 1 - \prod_{j=k+1}^{K-1} \frac{\pi_\theta(\boldsymbol{a}_j \mid \boldsymbol{s}_j)}{\pi_{\theta_{\mathrm{old}}}(\boldsymbol{a}_j \mid \boldsymbol{s}_j)} \right| \right] \\ &= \xi \cdot \sum_{k=1}^{K-1} \mathbb{E}_{\boldsymbol{s}_{\leq k} \sim \pi_{\theta_{\mathrm{old}}}} \left[\left| \frac{\pi_\theta(\boldsymbol{a}_k \mid \boldsymbol{s}_k)}{\pi_{\theta_{\mathrm{old}}}(\boldsymbol{a}_k \mid \boldsymbol{s}_k)} - 1 \right| \cdot \mathbb{E}_{\tau_{>k} \sim \pi_{\theta_{\mathrm{old}}}} \left[\left| 1 - \frac{\pi_{\theta,>k}(\tau_{>k} \mid \boldsymbol{s}_{k+1})}{\pi_{\theta_{\mathrm{old}},>k}(\tau_{>k} \mid \boldsymbol{s}_{k+1})} \right| \right] \right]. \end{aligned} \tag{19}$$

The inner expectation over future sub-trajectories is exactly twice the TV divergence between future trajectory distributions:

$$\mathbb{E}_{\tau_{>k} \sim \pi_{\theta_{\mathrm{old}}}} \left[\left| 1 - \frac{\pi_{\theta,>k}(\tau_{>k} \mid \boldsymbol{s}_{k+1})}{\pi_{\theta_{\mathrm{old}},>k}(\tau_{>k} \mid \boldsymbol{s}_{k+1})} \right| \right] = 2D_{\mathrm{TV}}\big(\pi_{\theta_{\mathrm{old}},>k}(\cdot \mid \boldsymbol{s}_{k+1}) \| \pi_{\theta,>k}(\cdot \mid \boldsymbol{s}_{k+1})\big).$$

Applying Theorem C.1 and bounding each term by $D_{\mathrm{TV}}^{\max}$:

$$D_{\mathrm{TV}}\big(\pi_{\theta_{\mathrm{old}},>k}(\cdot \mid \boldsymbol{s}_{k+1}) \| \pi_{\theta,>k}(\cdot \mid \boldsymbol{s}_{k+1})\big) \leq (K-1-k)\, D_{\mathrm{TV}}^{\max}(\pi_{\theta_{\mathrm{old}}} \| \pi_\theta).$$

Substituting back into Eq. 19:

$$\begin{aligned}
|\Delta(\pi_{\theta_{\mathrm{old}}}, \pi_\theta)| &\leq \xi \cdot \sum_{k=1}^{K-1} \mathbb{E}_{\boldsymbol{s}_k \sim \pi_{\theta_{\mathrm{old}}}} \left[\mathbb{E}_{\boldsymbol{a}_k \sim \pi_{\theta_{\mathrm{old}}}(\cdot|\boldsymbol{s}_k)} \left[\left| \frac{\pi_\theta(\boldsymbol{a}_k \mid \boldsymbol{s}_k)}{\pi_{\theta_{\mathrm{old}}}(\boldsymbol{a}_k \mid \boldsymbol{s}_k)} - 1 \right| \right] \right] \cdot 2(K-1-k)\, D_{\mathrm{TV}}^{\max} \\
&= 2\xi \cdot D_{\mathrm{TV}}^{\max} \sum_{k=1}^{K-1} (K-1-k) \cdot \mathbb{E}_{\boldsymbol{s}_k \sim \pi_{\theta_{\mathrm{old}}}} \left[2 D_{\mathrm{TV}}\big(\pi_{\theta_{\mathrm{old}}}(\cdot \mid \boldsymbol{s}_k) \| \pi_\theta(\cdot \mid \boldsymbol{s}_k)\big) \right] \\
&\leq 2\xi \cdot D_{\mathrm{TV}}^{\max} \sum_{k=1}^{K-1} (K-1-k) \cdot 2 D_{\mathrm{TV}}^{\max} \\
&= 4\xi \cdot D_{\mathrm{TV}}^{\max 2} \sum_{k=1}^{K-1} (K-1-k).
\end{aligned}$$

Evaluating the sum: $\sum_{k=1}^{K-1}(K-1-k) = \sum_{m=0}^{K-2} m = \frac{(K-1)(K-2)}{2}$. Therefore:

$$|\Delta(\pi_{\theta_{\mathrm{old}}}, \pi_\theta)| \leq 4\xi \cdot \frac{(K-1)(K-2)}{2} \cdot D_{\mathrm{TV}}^{\max 2} = 2\xi(K-1)(K-2) \cdot D_{\mathrm{TV}}^{\max 2}.$$

Substituting into the performance difference identity yields the desired bound:

$$J(\pi_\theta) - J(\pi_{\theta_{\mathrm{old}}}) \geq L'_{\theta_{\mathrm{old}}}(\pi_\theta) - 2\xi(K-1)(K-2) \cdot D_{\mathrm{TV}}^{\max}(\pi_{\theta_{\mathrm{old}}} \| \pi_\theta)^2.$$

This completes the proof. □

C.3 A Tighter Policy Improvement Bound

The quadratic dependence on the horizon K^2 in Theorem 3.2 can be overly pessimistic. By exploiting the fact that $D_{\mathrm{TV}} \leq 1$, we derive a tighter bound that is linear in K.

Starting from the intermediate step in Eq. 19, the inner expectation is $2D_{\mathrm{TV}}(\pi_{\theta_{\mathrm{old}},>k}(\cdot \mid \boldsymbol{s}_{k+1}) \| \pi_{\theta,>k}(\cdot \mid \boldsymbol{s}_{k+1}))$. Instead of applying Theorem C.1, we directly use the universal bound $D_{\mathrm{TV}} \leq 1$:

$$\begin{aligned}
|\Delta(\pi_{\theta_{\mathrm{old}}}, \pi_\theta)| &\leq \xi \cdot \sum_{k=1}^{K-1} \mathbb{E}_{\boldsymbol{s}_k \sim \pi_{\theta_{\mathrm{old}}}} \left[\mathbb{E}_{\boldsymbol{a}_k \sim \pi_{\theta_{\mathrm{old}}}(\cdot|\boldsymbol{s}_k)} \left[\left| \frac{\pi_\theta(\boldsymbol{a}_k \mid \boldsymbol{s}_k)}{\pi_{\theta_{\mathrm{old}}}(\boldsymbol{a}_k \mid \boldsymbol{s}_k)} - 1 \right| \right] \right] \cdot 2 \\
&= 4\xi \cdot \mathbb{E}_{\tau \sim \pi_{\theta_{\mathrm{old}}}} \left[\sum_{k=1}^{K-1} D_{\mathrm{TV}}\big(\pi_{\theta_{\mathrm{old}}}(\cdot \mid \boldsymbol{s}_k) \| \pi_\theta(\cdot \mid \boldsymbol{s}_k)\big) \right].
\end{aligned}$$

Combining both bounds, the policy improvement satisfies the composite guarantee:

$$J(\pi_\theta) - J(\pi_{\theta_{\mathrm{old}}}) \geq L'_{\theta_{\mathrm{old}}}(\pi_\theta) - \min\left(2\xi(K-1)(K-2) \cdot D_{\mathrm{TV}}^{\max 2},\ 4\xi \cdot \mathbb{E}_{\tau \sim \pi_{\theta_{\mathrm{old}}}} \left[\sum_{k=1}^{K-1} D_{\mathrm{TV},k} \right] \right),$$

where $D_{\mathrm{TV},k} = D_{\mathrm{TV}}(\pi_{\theta_{\mathrm{old}}}(\cdot \mid \boldsymbol{s}_k) \| \pi_\theta(\cdot \mid \boldsymbol{s}_k))$. The quadratic bound is tighter for small policy changes, while the linear bound is tighter for larger updates or longer horizons.

C.4 Connection to Gaussian Per-Step Divergence

For the Gaussian policies in Eq. 4, $\pi_{\theta_{\mathrm{old}}}(\cdot \mid \boldsymbol{s}_k) = \mathcal{N}(\boldsymbol{\mu}_{\theta_{\mathrm{old}}}, \sigma^2(t_k)\mathbf{I})$ and $\pi_\theta(\cdot \mid \boldsymbol{s}_k) = \mathcal{N}(\boldsymbol{\mu}_\theta, \sigma^2(t_k)\mathbf{I})$, the TV divergence admits the closed form:

$$D_{\mathrm{TV}}\big(\pi_{\theta_{\mathrm{old}}}(\cdot \mid \boldsymbol{s}_k) \| \pi_\theta(\cdot \mid \boldsymbol{s}_k)\big) = 2\Phi\left(\frac{\|\boldsymbol{\mu}_{\theta_{\mathrm{old}}} - \boldsymbol{\mu}_\theta\|}{2\sigma(t_k)} \right) - 1,$$

where Φ is the standard normal CDF. Since Φ is strictly monotonically increasing, the TV constraint $D_{\mathrm{TV}}^{\max} \le \delta$ is equivalent to:

$$\max_{\boldsymbol{s}_k} \|\boldsymbol{\mu}_{\theta_{\mathrm{old}}}(\boldsymbol{x}_{t_k}, t_k, \boldsymbol{c}) - \boldsymbol{\mu}_\theta(\boldsymbol{x}_{t_k}, t_k, \boldsymbol{c})\|^2 \le 4\sigma^2(t_k) \left[\Phi^{-1}\left(\frac{1+\delta}{2}\right)\right]^2 =: \delta'.$$

This establishes the equivalence between a *uniform* Gaussian mean-displacement constraint and a maximum-state TV constraint. Flow-DPPO evaluates the mean displacement at sampled states, so the equivalence holds pointwise on those states; obtaining the maximum-state premise in Theorem 3.2 would additionally require uniform state coverage. Appendix E.2 discusses this distinction.

D KL Divergence Between Gaussian Policies

In this section, we derive the KL divergence between old and new policies in flow models and establish its connection to the TV divergence used in the policy improvement bound.

D.1 General Gaussian KL Divergence

Let $p = \mathcal{N}(\boldsymbol{\mu}_1, \sigma^2\mathbf{I})$ and $q = \mathcal{N}(\boldsymbol{\mu}_2, \sigma^2\mathbf{I})$ be two isotropic Gaussians in $\mathbb{R}^d$ with the same covariance. The KL divergence is:

$$D_{\mathrm{KL}}(p\|q) = \frac{1}{2\sigma^2}\left[2(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\top \underbrace{\mathbb{E}_p[\boldsymbol{x} - \boldsymbol{\mu}_1]}_{=\mathbf{0}} + \|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|^2\right] = \frac{\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|^2}{2\sigma^2}. \tag{20}$$

Note that this is symmetric in the means: $D_{\mathrm{KL}}(p\|q) = D_{\mathrm{KL}}(q\|p)$ when the covariances are identical.

D.2 Connection Between KL and TV in the Gaussian Setting

For the same pair of Gaussians, the TV divergence is:

$$D_{\mathrm{TV}}(p, q) = 2\Phi\left(\frac{\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|}{2\sigma}\right) - 1.$$

Since both KL and TV are monotone functions of the single quantity $\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|/\sigma$, thresholding one is equivalent to thresholding the other. Specifically, the constraint $D_{\mathrm{TV}} \le \delta_{\mathrm{TV}}$ is equivalent to $\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|^2 \le 4\sigma^2[\Phi^{-1}((1+\delta_{\mathrm{TV}})/2)]^2$, which in turn is equivalent to $D_{\mathrm{KL}} \le 2[\Phi^{-1}((1+\delta_{\mathrm{TV}})/2)]^2$. This shows that the squared ℓ_2 distance $\|\boldsymbol{\mu}_{\theta_{\mathrm{old}}} - \boldsymbol{\mu}_\theta\|^2$ used in our mask is a unified divergence measure equivalent (up to monotone transformations) to both KL and TV divergences.

D.3 Application to Flow-SDE

For Flow-SDE (Eq. 2), the per-step policy is $\pi_\theta(\boldsymbol{x}_{t-\Delta t} \mid \boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{\mu}_\theta, \sigma_t^2 \Delta t \cdot \mathbf{I})$ where:

$$\boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t) = \boldsymbol{x}_t + \left[\boldsymbol{v}_\theta(\boldsymbol{x}_t, t) + \frac{\sigma_t^2}{2t}\big(\boldsymbol{x}_t + (1-t)\boldsymbol{v}_\theta(\boldsymbol{x}_t, t)\big)\right]\Delta t.$$

The difference in means is:

$$\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\theta_{\mathrm{old}}} = \left(1 + \frac{\sigma_t^2(1-t)}{2t}\right)\Delta t \cdot (\boldsymbol{v}_\theta - \boldsymbol{v}_{\theta_{\mathrm{old}}}).$$

Substituting into Eq. 20 with $\sigma^2 = \sigma_t^2 \Delta t$:

$$D_{\mathrm{KL}}^{\mathrm{SDE}}(\pi_{\theta_{\mathrm{old}}}\|\pi_\theta) = \frac{\Delta t}{2}\left(\frac{1}{\sigma_t} + \frac{\sigma_t(1-t)}{2t}\right)^2 \|\boldsymbol{v}_\theta(\boldsymbol{x}_t, t) - \boldsymbol{v}_{\theta_{\mathrm{old}}}(\boldsymbol{x}_t, t)\|^2. \tag{21}$$

D.4 Application to CPS

For CPS (Eq. 3), the policy mean is $\boldsymbol{\mu}_\theta^{\mathrm{CPS}} = (1-(t-\Delta t))\hat{\boldsymbol{x}}_0 + (t-\Delta t)\cos(\eta\pi/2)\hat{\boldsymbol{x}}_1$ and the variance is $\sigma_{\mathrm{CPS}}^2 = (t-\Delta t)^2 \sin^2(\eta\pi/2)$. Using $\hat{\boldsymbol{x}}_0 = \boldsymbol{x}_t - t\,\boldsymbol{v}_\theta$ and $\hat{\boldsymbol{x}}_1 = \boldsymbol{x}_t + (1-t)\boldsymbol{v}_\theta$, the difference in means is:

$$\boldsymbol{\mu}_\theta^{\mathrm{CPS}} - \boldsymbol{\mu}_{\theta_{\mathrm{old}}}^{\mathrm{CPS}} = [-(1-(t-\Delta t))t + (t-\Delta t)(1-t)\cos(\eta\pi/2)]\,(\boldsymbol{v}_\theta - \boldsymbol{v}_{\theta_{\mathrm{old}}}).$$

Let $c(t) = -(1-(t-\Delta t))t + (t-\Delta t)(1-t)\cos(\eta\pi/2)$. Then:

$$D_{\mathrm{KL}}^{\mathrm{CPS}}(\pi_{\theta_{\mathrm{old}}}\|\pi_\theta) = \frac{c(t)^2\|\boldsymbol{v}_\theta - \boldsymbol{v}_{\theta_{\mathrm{old}}}\|^2}{2(t-\Delta t)^2\sin^2(\eta\pi/2)}. \tag{22}$$

In previous work [38], the $2\sigma_{\mathrm{CPS}}^2$ normalization is dropped for numerical stability, reducing the divergence to $D(\pi_{\theta_{\mathrm{old}}}\|\pi_\theta) = \|\boldsymbol{\mu}_{\theta_{\mathrm{old}}}^{\mathrm{CPS}} - \boldsymbol{\mu}_\theta^{\mathrm{CPS}}\|^2$. We instead retain the full normalization in Eq. 22: because $\sigma_{\mathrm{CPS}}^2 \propto (t-\Delta t)^2$ shrinks at later denoising steps, the $\sigma_{\mathrm{CPS}}^{-2}$ factor amplifies the divergence where small velocity changes most affect the output, yielding a tighter constraint that prevents distribution collapse.

E Ratio Distribution and Scope of the Guarantee

E.1 Exact and Dimension-Averaged Ratios

At a sampled state, let the old and new per-step policies be $\pi_{\theta_{\mathrm{old}}} = \mathcal{N}(\boldsymbol{\mu}_{\theta_{\mathrm{old}}}, \sigma^2\mathbf{I})$ and $\pi_\theta = \mathcal{N}(\boldsymbol{\mu}_\theta, \sigma^2\mathbf{I})$ over n latent dimensions. Write $\boldsymbol{d} = \boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\theta_{\mathrm{old}}}$ and $\boldsymbol{a} = \boldsymbol{\mu}_{\theta_{\mathrm{old}}} + \sigma\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The exact joint ratio $r = \pi_\theta(\boldsymbol{a})/\pi_{\theta_{\mathrm{old}}}(\boldsymbol{a})$ satisfies

$$\log r = \frac{\boldsymbol{\epsilon}^\top \boldsymbol{d}}{\sigma} - \frac{\|\boldsymbol{d}\|^2}{2\sigma^2} \sim \mathcal{N}(-D_{\mathrm{total}}, 2D_{\mathrm{total}}), \qquad D_{\mathrm{total}} = \frac{\|\boldsymbol{d}\|^2}{2\sigma^2}. \tag{23}$$

Consequently,

$$\begin{aligned} \mathbb{E}[r] &= 1, & \mathbb{E}[\log r] &= -D_{\mathrm{total}}, \\ \mathrm{median}(r) &= e^{-D_{\mathrm{total}}}, & \Pr(r<1) &= \Phi\left(\sqrt{D_{\mathrm{total}}/2}\right). \end{aligned} \tag{24}$$

The median and probability bounds equal one and $1/2$, respectively, only on policy. Off policy, the distribution is right-skewed: more than half of its mass lies below one, while a thin right tail restores the exact unit mean. The skew is not Gaussian-specific; the equal-covariance Gaussian policy makes its magnitude available in closed form.

Flow-GRPO averages log density over the n latent dimensions before exponentiating. Its implemented ratio is therefore $\tilde{r} = e^L$, where $L = \log r/n$ and $\bar{D} = D_{\mathrm{total}}/n$:

$$\begin{aligned} L &\sim \mathcal{N}(-\bar{D}, 2\bar{D}/n), & \mathbb{E}[\tilde{r}] &= e^{-\bar{D}(1-1/n)}, \\ \mathrm{median}(\tilde{r}) &= e^{-\bar{D}}, & \Pr(\tilde{r}<1) &= \Pr(r<1). \end{aligned} \tag{25}$$

Thus dimension averaging slightly lowers the mean of the implemented ratio but does not alter which side of one a sample occupies. This distinction separates the exact joint-ratio identity $\mathbb{E}[r] = 1$ from the numerical ratio used by the implementation.

Table 3: Ratio statistics from the official Flow-GRPO implementation at an off-policy step. The mean log-ratio agrees with the measured per-dimension divergence to three significant figures.

$\mathrm{median}(\tilde{r})$	$\Pr(\tilde{r}<1)$	$\Pr(\tilde{r}>1)$	$\mathbb{E}[\tilde{r} \mid \tilde{r}<1]$	$\mathbb{E}[\tilde{r} \mid \tilde{r}>1]$	$\mathbb{E}[\log\tilde{r}]$	$\bar{D}$
0.9999977	**61.1%**	38.2%	0.9999919	1.0000061	-2.60×10^{-6}	2.60×10^{-6}

The remaining 0.7% of ratios are numerically equal to one. The below-one deviations are also larger on average (-8.1×10^{-6} versus $+6.1\times10^{-6}$ relative to one), which explains why sampled ratio clipping can behave asymmetrically even though the exact joint ratio has unit mean.

E.2 Group Normalization and Sampled-State Masking

Let $R_1, \ldots, R_G$ be rewards for independent samples from one prompt and let $\tilde{A}_i = R_i - G^{-1}\sum_j R_j$ denote the mean-centered advantage. Writing $\nabla J = \mathbb{E}[R\nabla\log\pi_\theta(x)]$, independence makes the cross terms in the group baseline vanish, whereas the self term contributes ∇J. Hence

$$\mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\tilde{A}_i\nabla\log\pi_\theta(x_i)\right] = \frac{G-1}{G}\nabla J. \tag{26}$$

The subsequent division by the group standard deviation is a positive scalar within each group, so it does not change that group's update direction. Because this scalar varies across groups, Eq. 26 is the exact constant-factor statement for mean centering, not a claim that the fully standardized estimator has one global constant scale.

The policy-improvement bound in Theorem 3.2 controls $\max_{\boldsymbol{s}} D_{\mathrm{TV}}(\pi_{\theta_{\mathrm{old}}}(\cdot \mid \boldsymbol{s}) \| \pi_\theta(\cdot \mid \boldsymbol{s}))$. In contrast, Flow-DPPO evaluates D_t only at states in the sampled rollout batch. Pinsker's inequality bounds TV at those checked states, but extending this pointwise statement to the maximum over all states requires a uniform coverage or generalization assumption that we do not claim. The mask is a conservative sample-wise filter: it can zero a surrogate term but never reverse the sign of a retained term. It therefore does not by itself establish global monotonic improvement.

E.3 Relation to Prior Divergence Masks and Broader Samplers

DPPO [27] motivates divergence-based trust regions for categorical language-model policies, where evaluating the full vocabulary distribution is expensive and Binary or Top-K reductions approximate the divergence. GRPO-Guard [39] retains sampled-ratio clipping and renormalizes the ratio distribution. Flow-DPPO instead uses the equal-covariance Gaussian structure to evaluate the per-step divergence in closed form and uses the sampled ratio only for the directional condition.

The closed-form Gaussian KL is a positive timestep-dependent weight multiplying the squared difference between the old and new predicted means. Removing that weight leaves an unweighted MSE, which remains density-free and can serve as a divergence proxy when a sampler has no tractable Gaussian KL; the threshold must be rescaled accordingly. This proxy does not make the KL exact for a genuinely non-Gaussian policy, and its empirical test is reported in Appendix H.4.3.

F Towards a Predictive Divergence Mask

We recall the asymmetric mask in Flow-DPPO (Eq. 18). The mask blocks the gradient (i.e., $M_t^i = 0$) when two conditions hold simultaneously: (i) the divergence $D_t > \delta$ already exceeds the trust-region threshold, and (ii) a directional condition signals that the optimization would push the policy *further away* from $\pi_{\theta_{\mathrm{old}}}$. Concretely, the directional condition triggers when $\hat{A}^i > 0 \wedge r_t^i > 1$ (the gradient would further increase an already-elevated ratio) or $\hat{A}^i < 0 \wedge r_t^i < 1$ (the gradient would further decrease an already-reduced ratio). These two cases can be compactly unified as:

$$M_t^i = 0 \quad \Longleftrightarrow \quad \mathrm{sgn}\big(\hat{A}^i \cdot (r_t^i - 1)\big) > 0 \quad \wedge \quad D_t > \delta. \tag{27}$$

While this design is effective in practice, the directional indicator $\mathrm{sgn}\big(\hat{A}^i(r_t^i - 1)\big)$ is a *heuristic proxy* for whether the upcoming gradient step will increase the divergence. In a ratio-based trust region (e.g., PPO clipping), this sign test is well-motivated: $r_t^i - 1$ directly reflects the deviation of the single-sample Monte Carlo estimate of the importance ratio, so the sign of $\hat{A}^i(r_t^i - 1)$ faithfully indicates whether the surrogate objective would drive the ratio further from unity. However, in a *divergence-based* trust region where the constraint is on $D_t = D_{\mathrm{KL}}(\pi_{\theta_{\mathrm{old}}} \| \pi_\theta)$, the connection is less direct. The ratio r_t^i is a stochastic quantity evaluated at a single sampled action, whereas D_t measures a distributional distance that integrates over all actions. A positive $\hat{A}^i(r_t^i - 1)$ does not guarantee that the gradient step will increase D_t, nor does a negative value guarantee a decrease.

In this section we exploit the Gaussian structure of flow model policies to derive a more principled masking criterion. We first predict how a single gradient step changes D_t (§F.1), obtaining a closed-form expression that decomposes into a first-order directional term and a second-order magnitude term. The sign of the first-order term yields an exact directional criterion $\mathrm{sgn}\big(\hat{A} \cdot (\log r_t - D_t)\big)$, which recovers the current sign test in the small-divergence regime but reveals a correction when the policy has already drifted. The full expression further accounts for the step size and gradient magnitude, leading to a predictive mask (§F.2) that blocks predicted threshold crossings and further outward drift.

F.1 Predicting Post-Update Divergence

Fix a denoising step with state $\boldsymbol{x}_t$ and suppress the time index for brevity. Write $\boldsymbol{\mu} \equiv \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t)$, $\boldsymbol{\mu}_{\text{old}} \equiv \boldsymbol{\mu}_{\theta_{\text{old}}}(\boldsymbol{x}_t, t)$, $\boldsymbol{d} = \boldsymbol{\mu} - \boldsymbol{\mu}_{\text{old}}$, and $D_t = \|\boldsymbol{d}\|^2/(2\sigma^2)$. The sampled action is $\boldsymbol{x}_{t-\Delta t} = \boldsymbol{\mu}_{\text{old}} + \sigma\boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

We derive how a single gradient step on the surrogate objective $L = r_t \cdot \hat{A}$ changes the divergence D_t. The policy gradient with respect to $\boldsymbol{\mu}$ is:

$$\nabla_{\boldsymbol{\mu}} L = \hat{A} \cdot r_t \cdot \nabla_{\boldsymbol{\mu}} \log r_t = \frac{\hat{A} \cdot r_t}{\sigma^2}(\sigma\boldsymbol{\epsilon} - \boldsymbol{d}).$$

With effective learning rate ρ, the updated mean is $\boldsymbol{\mu}_{\text{new}} = \boldsymbol{\mu} + \rho \cdot \nabla_{\boldsymbol{\mu}} L$. Let $\boldsymbol{g} = \sigma\boldsymbol{\epsilon} - \boldsymbol{d}$. The predicted post-update divergence is:

$$\begin{aligned} D_t^{\text{new}} &= \frac{\|\boldsymbol{\mu}_{\text{new}} - \boldsymbol{\mu}_{\text{old}}\|^2}{2\sigma^2} = \frac{1}{2\sigma^2}\left\|\boldsymbol{d} + \frac{\rho\,\hat{A}\,r_t}{\sigma^2}\,\boldsymbol{g}\right\|^2 \\ &= D_t + \frac{\rho\,\hat{A}\,r_t}{\sigma^4}\,\boldsymbol{g}^\top\boldsymbol{d} + \frac{\rho^2\hat{A}^2 r_t^2}{2\sigma^6}\|\boldsymbol{g}\|^2. \end{aligned} \tag{28}$$

From the ratio decomposition (Eq. 13), $\log r_t = \boldsymbol{\epsilon}^\top\boldsymbol{d}/\sigma - \|\boldsymbol{d}\|^2/(2\sigma^2)$, which gives $\boldsymbol{g}^\top\boldsymbol{d} = \sigma^2(\log r_t - D_t)$. The first-order term thus simplifies to $(\rho\,\hat{A}\,r_t/\sigma^2)(\log r_t - D_t)$. The three terms in Eq. 28 have clear interpretations: (1) the current divergence D_t; (2) a first-order term whose sign determines whether the gradient step increases or decreases the divergence; (3) a non-negative second-order term that grows with the step size ρ and gradient magnitude $\|\boldsymbol{g}\|$, always contributing positively to D_t^{new}.

The first-order directional criterion. The direction of divergence change is mainly determined by the sign of the first-order term. Since $r_t > 0$ and $\rho > 0$, this sign equals:

$$\operatorname{sgn}\Big(\hat{A} \cdot \big(\log r_t - D_t\big)\Big). \tag{29}$$

When this is positive, an infinitesimal gradient step increases D_t to first order; when negative, it decreases D_t to first order. Equivalently, this is the sign of the inner product $\langle \nabla_{\boldsymbol{\mu}} L, \nabla_{\boldsymbol{\mu}} D_t \rangle$, confirming that the surrogate gradient projects onto the divergence-increasing direction.

Recovery of the current mask. In the small-divergence regime $D_t \ll 1$ (which is the typical operating range when the trust region is effective), the correction $D_t \approx 0$ and the criterion simplifies to $\operatorname{sgn}(\hat{A} \cdot \log r_t)$. Since $\operatorname{sgn}(\log r_t) = \operatorname{sgn}(r_t - 1)$, this is equivalent to $\operatorname{sgn}\big(\hat{A} \cdot (r_t - 1)\big)$, which is exactly the directional condition in Eq. 27. Thus, the current Flow-DPPO mask implements the correct first-order divergence-increasing criterion in this regime.

The correction term. When D_t is non-negligible (i.e., the policy has already drifted appreciably), the true divergence-change direction is $\operatorname{sgn}\big(\hat{A} \cdot (\log r_t - D_t)\big)$ rather than $\operatorname{sgn}\big(\hat{A} \cdot (r_t - 1)\big)$. The subtracted term D_t shifts the decision boundary: a sample must have $\log r_t > D_t > 0$ (rather than merely $\log r_t > 0$) before the positive-advantage gradient is classified as divergence-increasing. Intuitively, when the policy has already moved away from $\pi_{\theta_{\text{old}}}$, a moderately elevated ratio does not necessarily push it further; only sufficiently large ratios do. This yields a first natural refinement of the mask: replacing $\operatorname{sgn}\big(\hat{A}(r_t - 1)\big)$ with $\operatorname{sgn}\big(\hat{A} \cdot (\log r_t - D_t)\big)$ as the directional indicator, which we call the *first-order predictive mask*:

$$M_t^{(1)} = \begin{cases} 0, & \text{if } \operatorname{sgn}\Big(\hat{A} \cdot \big(\log r_t - D_t\big)\Big) > 0 \ \wedge \ D_t > \delta, \\ 1, & \text{otherwise.} \end{cases} \tag{30}$$

This mask uses only quantities already computed during training ($\hat{A}$, r_t, D_t) and requires no additional hyperparameters beyond the existing threshold δ.

F.2 The Predictive Mask

Based on Eq. 28, we define the *(full) predictive mask* that blocks an update only when the predicted post-update divergence exceeds both the current divergence and the trust-region threshold:

$$M_t^{\text{pred}} = \begin{cases} 0, & \text{if } D_t^{\text{new}} > \max\{D_t, \delta\}, \\ 1, & \text{otherwise.} \end{cases} \tag{31}$$

Comparison with the first-order mask. The first-order mask (Eq. 30) uses the local direction of change and fires only after the current divergence already exceeds δ. The full predictive mask instead evaluates the finite predicted step. If $D_t \le \delta$, it blocks only an update predicted to cross the threshold. If $D_t > \delta$, it blocks only when $D_t^{\text{new}} > D_t$, while retaining any pull-back update with $D_t^{\text{new}} \le D_t$ even if the sample remains outside the trust region. The non-negative second-order term in Eq. 28 also allows the full mask to detect a large step that overshoots despite a safe first-order direction.

Recovery of the existing mask. Let $c_t = (\hat{A} r_t/\sigma^2)(\log r_t - D_t)$, so that $D_t^{\text{new}} - D_t = \rho c_t + \mathcal{O}(\rho^2)$. When $D_t > \delta$, the predictive condition becomes $D_t^{\text{new}} > D_t$. For sufficiently small $\rho > 0$ and nonzero c_t, this is equivalent to $c_t > 0$, recovering the first-order predictive mask. In the additional small-divergence regime, $\operatorname{sgn}(c_t)$ reduces to $\operatorname{sgn}\big(\hat{A}(r_t - 1)\big)$, recovering the directional condition of the current Flow-DPPO mask. When $D_t \le \delta$, the full mask adds a proactive threshold-crossing check that is absent from the current reactive mask.

F.3 Discussion on Mask Variants

Relation among mask variants. The directional criteria form successive approximations rather than nested acceptance sets:

$$\underbrace{\operatorname{sgn}\big(\hat{A}(r_t - 1)\big)}_{\text{current (Eq. 27)}} \xrightarrow[D_t \ll 1]{\text{refine}} \underbrace{\operatorname{sgn}\big(\hat{A}(\log r_t - D_t)\big)}_{\text{first-order (Eq. 30)}} \xrightarrow[\text{finite step}]{\text{predict}} \underbrace{D_t^{\text{new}} > \max\{D_t, \delta\}}_{\text{full predictive (Eq. 31)}}.$$

The current mask is the cheapest and suffices when the trust region keeps D_t small throughout training. The first-order mask refines the local directional decision with no additional hyperparameters. The full predictive mask additionally requires an effective learning rate estimate and uses the predicted finite-step divergence to preserve pull-back updates while preventing threshold crossings and further outward drift.

Local approximation. The analysis treats $\boldsymbol{\mu}$ as a free vector, whereas in practice it is the output of a neural network. The actual change in $\boldsymbol{\mu}(\boldsymbol{x}_t, t)$ is coupled to changes at all other inputs through shared parameters. The predictive mask is thus a local approximation that is most accurate when the effective learning rate is small and the network Jacobian is approximately preserved across one step.

We leave empirical validation of the predictive masks to future work. The key contribution of this analysis is twofold: it provides a theoretical justification for the existing asymmetric condition (showing it is the correct first-order criterion in the small-divergence regime), and it charts a principled path toward more refined trust-region enforcement that exploits the Gaussian structure of flow model policies.

G Experimental Details

G.1 Computational Resources.

All experiments are conducted on NVIDIA H20 96GB GPUs. The main results in Table 1 require approximately 90K GPU hours in total (across SD3.5, FLUX2-klein-base-9B, and FLUX1-dev with all methods and reward configurations). Including all ablation studies, multi-epoch experiments, and auxiliary runs, the overall computational cost for all experiments reported in this paper is approximately 140K GPU hours.

G.2 Hyperparameters.

LoRA is used for all models. We use LoRA $r = 32$ and $\alpha = 64$ for SD3.5, $r = 64$ and $\alpha = 128$ for FLUX2-9B and FLUX.1-dev. The learning rate is set to 3×10^{-4} for all models aligning to previous works. We set the training resolution to 512×512, number of denoising steps to 10 for SD3.5 and 14 for FLUX2-9B.

For GRPO setting, we use group size 16 and number of groups 64 per epoch for all methods. The PPO clip threshold is set to 1×10^{-4} for Flow-GRPO and Flow-CPS, and 4×10^{-6} for GRPO-Guard, following the official recommendation. The thresholds for KL-clipping are set to 1×10^{-7} for Flow-DPPO and 1×10^{-6} for Flow-DPPO+CPS due to their different KL-scaling factors. We applied

the strategy proposed in MixGRPO [15] for faster convergence and better performance. Specifically, we mix ODE and SDE sampling and randomly select 3 steps out of first half of the denoising steps for SDE sampling. The noise level for SDE sampling (η in CPS sampling) is set to 0.8.

For Diffusion-NFT, we follow the official implementation for SD3.5 for the rest of the hyperparameters, such as EMA schedule.

G.3 Training and Evaluation Protocol

For the image experiments, we use AdamW with constant learning rate 3×10^{-4}, training seed 42, 64 prompt groups, group size 16, and therefore 1024 rollouts per epoch, following Flow-GRPO. We report fixed final saved checkpoints (epoch 3000 for SD3.5 and epoch 1000 for FLUX2-9B and FLUX.1-dev) without test-reward checkpoint selection. Both log probability and Gaussian KL are means over latent dimensions; the ratio is therefore a per-dimension geometric mean and the KL threshold is on the same per-dimension scale.

For the video experiment in Figure 3, both methods use Wan2.1-T2V-1.3B with LoRA rank 64, learning rate 10^{-4}, seed 42, CPS sampling with $\eta = 0.9$, and 1024 videos per epoch. Training rollouts contain 21 frames at 512^2 with 16 denoising steps; evaluation uses 41 frames at 720^2, 50 denoising steps, and VideoReward, every 100 epochs. The mask is the only algorithmic difference ($\epsilon = 10^{-4}$ clipping versus $\delta = 10^{-6}$ asymmetric KL masking); "Flow-GRPO" therefore denotes ratio clipping under the same CPS sampler.

H Additional Experimental Results

This section complements Section 4. Unless stated otherwise, each comparison changes only the trust-region mechanism and retains the model, sampler, data, reward, and training budget of the matched paper setting. In the PickScore and Gaussian-KL penalty comparisons, both methods use CPS sampling; there, "Flow-GRPO" denotes the ratio-clipping objective under the same CPS sampler.

H.1 Full Results on Image Generation

H.1.1 Quantitative Summary on GenEval2

Tables 4 and 5 report the end-of-training Soft TIFA_{GM} score on GenEval2 for each method. Table 4 additionally reports end-of-training ancillary CLIP, PickScore, and HPSv2 rewards on both the in-domain GenEval2 prompt set and the held-out out-of-domain PickScore validation prompts, contextualizing both SD3.5-medium and FLUX2-klein-base-9B by stacking six blocks: published reference numbers for state-of-the-art text-to-image systems, the corresponding pretrained-baseline scores (no RL), and the five RL fine-tuning algorithms applied to each base model under both the single-reward (GenEval2-only) and multi-reward (GenEval2 + CLIP + PickScore + HPSv2) configurations. Table 5 then expands the per-method Soft TIFA_{GM} comparison to all training settings reported in this paper.

H.1.2 Additional Training Curves

We provide the training curves on SD3.5 for the single-reward setting in Figure 7 (the multi-reward setting is in Figure 2 in the main body). We also provide the FLUX2-9B multi-reward training curves in Figure 8 and FLUX.1-dev in Figure 9.

H.2 KL Divergence Curves

Figure 10 visualizes the per-step KL divergence between the current and reference (pre-trained) model across all six training settings and two SDE schedules. The corresponding end-of-training values are reported in Table 2 of the main body.

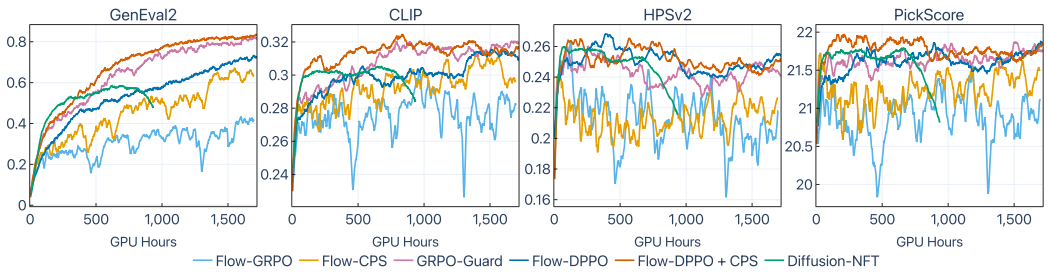


Figure 7: Training curves on SD3.5 for single-reward setting, including Diffusion-NFT [43] as an additional baseline. Flow-DPPO variants achieve state-of-the-art performance and less catastrophic forgetting on out-of-domain rewards, consistent with the main results.

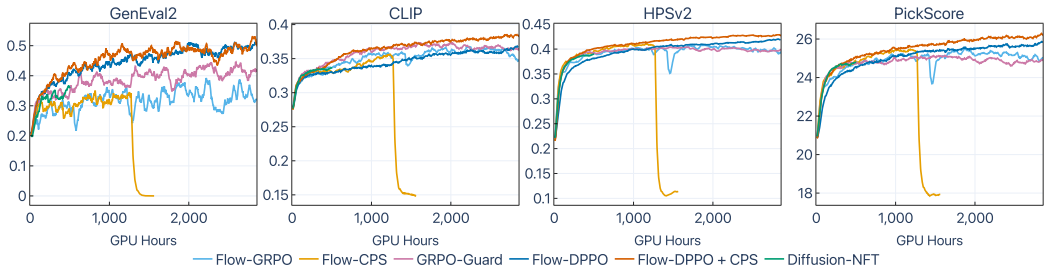


Figure 8: Training curves on FLUX2-9B for multi-reward setting (GPU hours). Flow-DPPO variants consistently outperform the baselines across all metrics, with a notable improvement on the GenEval2 reward.

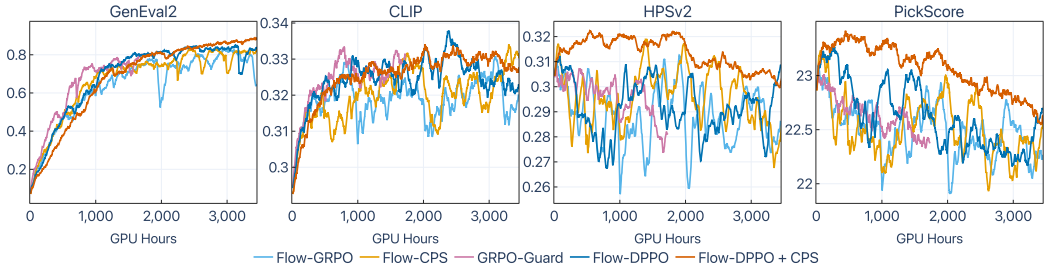


Figure 9: Training curves on FLUX.1-dev for single-reward setting.

Table 4: GenEval2 [Soft TIFA_{GM}, defined in [12]] together with ancillary CLIP, PickScore, and HPSv2 rewards at the end of training. The four in-domain columns are evaluated on the GenEval2 prompt set (the official released evaluation set of 800 prompts); the three out-of-domain columns are evaluated on the PickScore prompt set. Within each RL block, **bold** marks the per-column top-1 method and underline the per-column top-2 method. Blue rows highlight our two contributions.

	In-Domain (GenEval2)				Out-of-Domain (PickScore)		
Model	GenEval2	CLIP	PickScore	HPSv2	CLIP	PickScore	HPSv2
State-of-the-Art T2I Models							
SD3.5-large	22.8	–	–	–	–	–	–
Bagel + CoT	23.1	–	–	–	–	–	–
Qwen-Image	33.8	–	–	–	–	–	–
Gemini 2.5 Flash Image	44.6	–	–	–	–	–	–
Pretrained baselines (before RL)							
SD3.5-medium	12.4	0.250	21.00	0.213	0.244	19.99	0.210
FLUX2-klein-base-9B	25.4	0.281	20.92	0.228	0.254	20.05	0.230
FLUX.1-dev	23.3	0.297	23.26	0.315	0.276	21.91	0.304
SD3.5-medium, single-reward RL fine-tuning							
Flow-GRPO	56.6	0.297	21.21	0.219	0.252	19.33	0.206
Flow-CPS	74.8	0.313	21.68	0.235	0.260	19.94	0.220
GRPO-Guard	**85.8**	**0.328**	22.03	0.252	0.265	19.94	0.214
Diffusion-NFT	64.5	0.307	21.69	0.251	0.262	20.24	0.239
Flow-DPPO	78.9	0.319	**22.06**	**0.263**	0.265	20.45	**0.253**
Flow-DPPO + CPS	84.1	0.316	21.99	0.262	**0.272**	**20.50**	0.246
SD3.5-medium, multi-reward RL fine-tuning							
Flow-GRPO	39.9	0.358	25.09	0.399	0.273	22.07	0.349
Flow-CPS	44.6	0.359	25.51	0.407	0.265	22.08	0.343
GRPO-Guard	47.8	0.353	25.64	0.409	0.272	22.32	0.354
Diffusion-NFT	42.5	0.334	25.30	0.394	0.269	22.52	0.355
Flow-DPPO	48.1	0.345	25.63	0.409	0.273	**22.58**	0.360
Flow-DPPO + CPS	**51.6**	**0.369**	**25.72**	**0.415**	**0.279**	22.51	**0.361**
FLUX2-klein-base-9B, single-reward RL fine-tuning							
Flow-GRPO	84.5	0.314	21.82	0.276	0.264	20.84	0.280
Flow-CPS	82.7	0.311	21.82	0.261	0.275	21.15	0.267
GRPO-Guard	82.8	0.312	20.52	0.210	0.230	18.45	0.167
Flow-DPPO	85.1	**0.331**	**22.22**	**0.294**	**0.278**	**21.27**	**0.285**
Flow-DPPO + CPS	**92.6**	0.315	21.97	0.279	0.265	20.79	0.272
FLUX2-klein-base-9B, multi-reward RL fine-tuning							
Flow-GRPO	46.8	0.371	25.61	0.412	0.277	22.62	0.357
Flow-CPS	47.1	0.361	25.70	0.416	0.276	22.85	0.364
GRPO-Guard	49.0	0.375	25.27	0.411	0.269	21.99	0.349
Diffusion-NFT	47.3	0.336	24.87	0.389	0.274	22.47	0.351
Flow-DPPO	**57.7**	0.364	25.76	0.418	0.282	22.90	0.368
Flow-DPPO + CPS	55.2	**0.386**	**26.15**	**0.427**	**0.287**	**22.97**	**0.370**
FLUX.1-dev, single-reward RL fine-tuning							
Flow-GRPO	87.8	0.331	23.03	0.311	**0.291**	21.85	**0.311**
Flow-CPS	91.2	0.328	23.20	0.317	0.288	**21.98**	0.307
GRPO-Guard	87.6	**0.333**	22.69	0.293	0.286	21.03	0.276
Flow-DPPO	90.7	0.331	23.15	**0.323**	0.290	21.60	0.300
Flow-DPPO + CPS	**91.6**	0.331	**23.29**	0.322	0.289	21.91	0.305

Table 5: End-of-training Soft TIFA_{GM} on GenEval2 (%) across six training configurations (columns) and five RL algorithms (rows). The six columns correspond, left-to-right, to Figs. 2, 8, 12, 7, 2, 9. Per-column **bold** and underline mark the top-1 and top-2 methods; blue rows highlight our two contributions.

	FLUX2-9B			SD3.5		FLUX.1-dev
Method	Single	Multi	+CFG	Single	Multi	Single
Flow-GRPO	84.5	46.8	54.6	56.6	39.9	87.8
Flow-CPS	82.7	47.1	89.0	74.8	44.6	91.2
GRPO-Guard	82.8	49.0	78.8	**85.8**	47.8	87.6
Diffusion-NFT	–	47.3	–	64.5	42.5	–
Flow-DPPO	85.1	**57.7**	87.4	78.9	48.1	90.7
Flow-DPPO + CPS	**92.6**	55.2	**91.0**	84.1	**51.6**	**91.6**

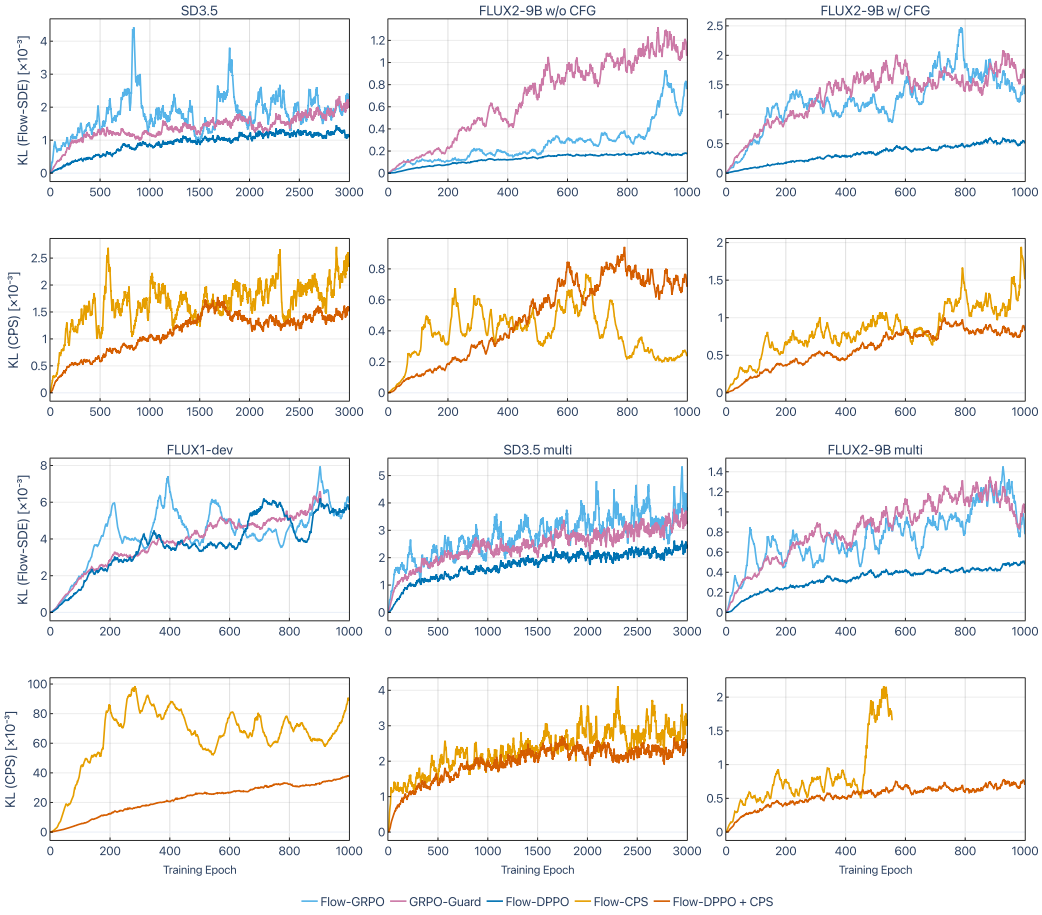


Figure 10: KL-divergence between the current and reference (pre-trained) model during training, across six training settings (columns: four single-reward — SD3.5, FLUX2-9B w/o CFG, FLUX2-9B w/ CFG, FLUX.1-dev; two multi-reward — SD3.5 multi, FLUX2-9B multi) and two SDE schedules (rows: Flow-SDE, CPS). For each schedule, Flow-DPPO variants maintain a lower KL divergence with the pre-trained model, indicating less catastrophic forgetting and reward hacking. The only exception is in the FLUX2-9B w/o CFG setting under the CPS schedule, where Flow-DPPO + CPS shows a higher KL divergence than the Flow-CPS baseline after about epoch 500. The Flow-CPS run on FLUX2-9B multi collapsed at epoch 480; we plot its full logged trajectory and report its end-of-training KL at the run's last logged step in Table 2.

H.3 Robustness of the Main Results

H.3.1 Variation Across Training and Evaluation Seeds

We repeat two SD3.5 single-reward settings with two training seeds. The table reports mean and standard deviation over $n = 2$ runs; this small sample measures observed run variation and is not used for a significance claim.

Table 6: Variation across two training seeds. Only GenEval2 is optimized; HPSv2, PickScore, and CLIP are unoptimized evaluation metrics.

Model and sampler	GenEval2	HPSv2	PickScore	CLIP	GenEval2 gain over clipping
SD3.5 + CPS	$\mathbf{0.8442 \pm 0.0050}$	0.2594 ± 0.0030	21.97 ± 0.03	0.3200 ± 0.0055	$+0.093$
SD3.5 + Flow-SDE	$\mathbf{0.7886 \pm 0.0007}$	0.2598 ± 0.0051	21.99 ± 0.10	0.3210 ± 0.0032	$+0.223$

We also reevaluate the final SD3.5 checkpoint on all 800 official prompts with five generation seeds $\{43, \ldots, 47\}$. The per-seed GenEval2 scores are $0.845, 0.853, 0.851, 0.861, 0.856$, giving $\mathbf{0.8532 \pm 0.0052}$; the value 0.8576 reported by the original run lies within one standard deviation.

H.3.2 A Second Preference Reward

To test a reward distinct from GenEval2, we directly optimize PickScore for 4000 epochs. The Flow-GRPO and Flow-DPPO runs have identical scheduler, data, model, evaluation, reward, and compute configurations and differ only in the clipping versus asymmetric KL mask.

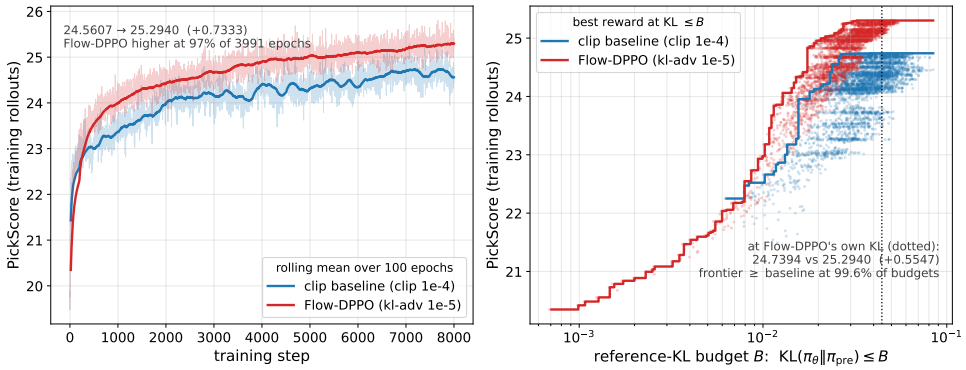


Figure 11: PickScore-only optimization. Left: the 100-epoch rolling mean of the on-policy training reward increases from 24.5607 with clipping to 25.2940 with Flow-DPPO ($+0.7333$); Flow-DPPO is higher at 97% of 3991 matched epochs. Right: reward under an oracle reference-KL budget; at Flow-DPPO's final reference KL 4.43×10^{-2}, the oracle-stopped baseline reaches 24.7394 versus 25.2940 ($+0.5547$). The final clipping reference KL is 5.2×10^{-2}, so Flow-DPPO ends approximately 15% lower.

The on-policy training reward is the optimized objective reported above, whereas the held-out PickScore evaluation saturates near its upper range and changes from 24.4873 for clipping to 24.5408 for Flow-DPPO. This experiment shows that the gain is not specific to the GenEval2 reward used by the main single-reward experiments.

H.3.3 Classifier-Free Guidance

Previous works found that CFG heavily affects the training convergence and performance [43]. Here, we study the effect of CFG on the training of Flow-DPPO on FLUX2-9B, as shown in Figure 12, where the CFG scale is set to 4.0 following the official recommendation. With CFG, Flow-DPPO variants still achieve state-of-the-art performance on the training reward (GenEval2) and mitigate catastrophic forgetting on the out-of-domain prompts, consistent with the observations in previous discussions. This shows that the divergence-based mask is robust under CFG and continues to deliver strong performance.

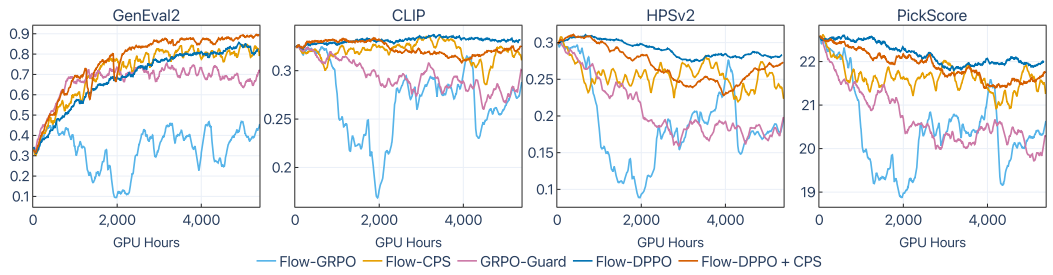


Figure 12: Training curves on FLUX2-9B with CFG scale 4.0. Flow-DPPO variants remain robust under CFG, achieving strong performance with less catastrophic forgetting.

H.4 Analysis of the Divergence Mask

H.4.1 Which Updates Stabilize Training?

We split each off-policy step into: (1) $D \leq \delta$, inside the trust region; (2) $D > \delta$ with the update pulling the sample back; and (3) $D > \delta$ with the update pushing the sample farther out. Flow-DPPO keeps groups (1) and (2) and drops group (3). The right panel of Figure 4 in the main text shows the CPS-matched runs; Table 7 and Figure 13 additionally include a pure on-policy run without any trust region.

Table 7: Directional ablation over the first 1000 epochs. The collapse column records the first observed collapse event; the extended trajectories are shown in Figure 13.

Updates retained	Peak reward (epoch)	Collapse event (epoch)
Pure on-policy, no trust region	0.317 (394)	464
Group (1), without asymmetry	0.295 (405)	443
Group (2), $D > \delta$ pull-back	0.545 (982)	–
Groups (1)+(2), Flow-DPPO	**0.682 (990)**	–

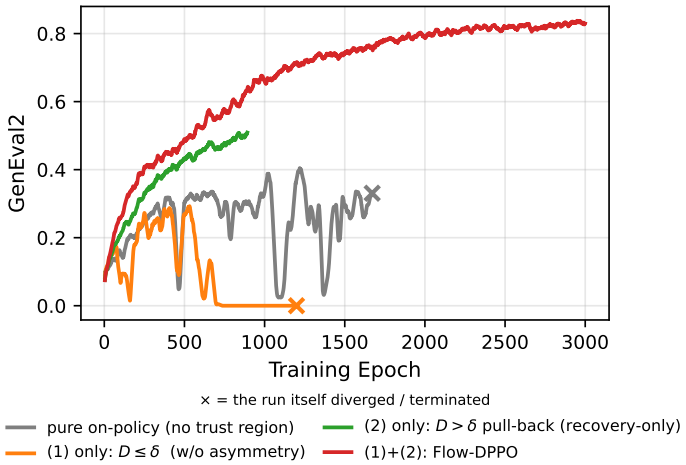


Figure 13: Trust-region decomposition on SD3.5 with single-reward GenEval2 ($\delta = 10^{-6}$); curves are 25-epoch rolling means of training reward. All four runs reach epoch 892: pure on-policy, group (1), group (2), and Flow-DPPO attain 0.299, 0.000, 0.509, and 0.635, respectively. Group (1) collapses to zero, whereas group (2) remains stable but below Flow-DPPO, showing that the pull-back updates provide stability while in-region updates supply reward progress. The pure on-policy run uses the Flow-SDE schedule; the other runs use CPS. The group-(2) run was manually stopped after approximately 1000 epochs rather than failing.

H.4.2 Multi-Sample Divergence Estimation

For K probes $\boldsymbol{z}_k \sim \pi_{\theta_{\text{old}}}(\cdot \mid \boldsymbol{x}_t)$, the Monte Carlo estimator

$$\widehat{D}_K = -\frac{1}{K}\sum_{k=1}^{K}\left[\log \pi_\theta(\boldsymbol{z}_k) - \log \pi_{\theta_{\text{old}}}(\boldsymbol{z}_k)\right], \qquad \mathbb{E}[\widehat{D}_K] = D_{\text{KL}}(\pi_{\theta_{\text{old}}} \| \pi_\theta), \tag{32}$$

is an unbiased but noisy estimate of the same divergence used by Flow-DPPO; the closed form is its zero-estimation-noise limit. We evaluate it at the scale of the SD3.5 512^2 ablation ($n = 65{,}536$, $\bar{D} = 1.44 \times 10^{-6}$, $\delta = 10^{-7}$).

Table 8: Noise of the multi-sample KL estimator and agreement with decisions made by the exact mask. Noise is the estimator standard deviation divided by D.

K	Noise of $\widehat{D}_K$	Agreement with exact mask
1	473%	58.2%
4	237%	66.0%
16	119%	79.0%
32	84%	86.9%
64	59%	93.5%
Closed form	**0 (exact)**	**100%**

Table 9: End-to-end training rewards when mask decisions use the closed-form KL or $K = 64$ Monte Carlo probes.

Mask divergence	Ep. 200	Ep. 400	Ep. 600
Closed-form KL	0.248	0.372	0.441
$K = 64$ estimate	0.250	0.364	0.426

At $K = 64$, the estimator reproduces 93.5% of exact-mask decisions and tracks the closed-form training trajectory. Accurate masking decisions, rather than a manually altered ratio, account for the similar behavior.

H.4.3 Generality Beyond the Timestep-Weighted KL

For an equal-covariance Gaussian transition, the closed-form KL is a positive timestep-dependent weight multiplying the MSE between the old and new predicted means. We test the density-free proxy from Appendix E.3 by removing this weight and rescaling the threshold. Both runs use SD3.5 with CPS, PickScore as the single reward, the PickScore prompt set, and identical seeds and configurations.

Table 10: CPS timestep-weighted KL versus an unweighted MSE proxy. Entries are PickScore training rewards except for the final evaluation column.

Mask divergence	Ep. 1000	Ep. 2000	Ep. 2400	Eval. at 2400
Timestep-weighted KL	24.18	24.64	**24.77**	**24.20**
Unweighted MSE	24.21	24.62	24.73	24.11
Difference	-0.02	$+0.02$	$+0.04$	$+0.09$

The two runs differ by at most 0.09 on the reported measurements. This experiment supports an MSE proxy when the Gaussian weighting is unavailable; it does not establish exact KL control for an arbitrary non-Gaussian policy.

H.5 Alternative Forms of KL Control

H.5.1 Exact Gaussian-KL Penalty

We replace the mask by a soft penalty on the same per-dimension old–new Gaussian KL $D_t = D_{\text{KL}}(\pi_{\theta_{\text{old}}} \| \pi_\theta)/n$ while holding all other settings fixed. A run is called collapsed at the first epoch after its peak where reward falls below half of that peak.

Table 11: Direct Gaussian-KL penalty baseline on SD3.5 with CPS and single-reward GenEval2.

Trust-region mechanism	Peak smoothed reward	Collapse epoch
Flow-DPPO KL mask	**0.841**	–
Flow-GRPO ratio clip	0.684	–
Gaussian-KL penalty, $\beta_{\mathrm{KL}} = 10^{-3}$	0.311	229
Gaussian-KL penalty, $\beta_{\mathrm{KL}} = 10^{-2}$	0.216	228

Both penalty runs peak below 0.32 and collapse near epoch 230, whereas ratio clipping and the KL mask train for the full 3000 epochs. A soft penalty therefore does not substitute for the mask in this setting.

H.5.2 Oracle KL Early Stopping

For every reference-KL budget B, we select the checkpoint with the best GenEval2 among all evaluated checkpoints satisfying $D_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{pre}}) \leq B$. This grants each method an oracle choice of training horizon; HPSv2 is evaluated at that same GenEval2-selected checkpoint.

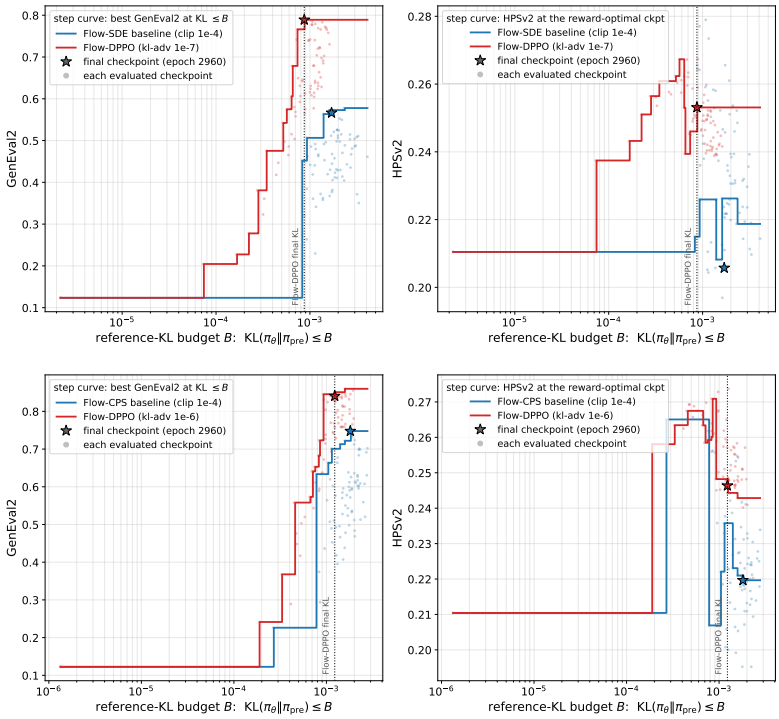


Figure 14: Reward–reference-KL Pareto curves for post-hoc oracle stopping (top: Flow-SDE; bottom: CPS). Under Flow-DPPO's final KL budget, it improves GenEval2 over the oracle-stopped clipping baseline by $+0.337$ for Flow-SDE (0.789 versus 0.452 at $B = 8.7 \times 10^{-4}$) and $+0.140$ for CPS (0.841 versus 0.701 at $B = 1.2 \times 10^{-3}$). Flow-DPPO's frontier is no worse at all 148 informative budgets in each exported comparison; faint dots are evaluated checkpoints and stars are final checkpoints.

Early stopping only chooses when to leave an existing trajectory; it does not control off-policy drift while optimizing. Even with oracle checkpoint selection, the clipping baseline remains below Flow-DPPO at matched reference-KL budgets.

H.5.3 Reference KL Regularization Strength

We ablate the strength of the $D_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{ref}})$ regularization term (controlled by β) on FLUX2-klein-base-9B under the multi-reward GDPO setting with CPS scheduling. Figure 15 shows the training reward curves and Figure 16 shows the KL divergence from the pretrained model. A moderate

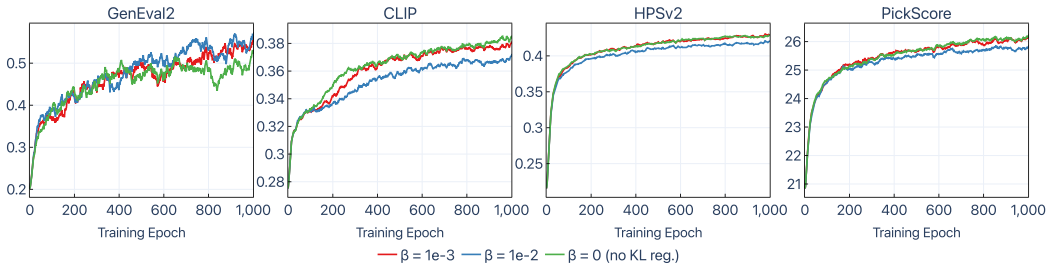


Figure 15: Training reward curves under three $D_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{ref}})$ regularization strengths (β) on FLUX2-klein-base-9B (multi-reward GDPO, CPS schedule). A moderate $\beta{=}10^{-3}$ suppresses early reward hacking on PickScore and HPSv2, balancing cross-reward gradients and boosting final GenEval2 performance without hurting end-of-training performance on any individual reward.

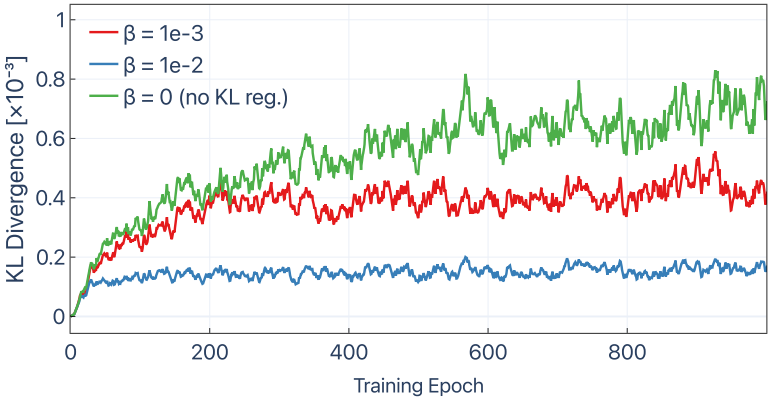


Figure 16: $D_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{ref}})$ during training for different β settings.

regularization strength ($\beta{=}10^{-3}$) further mitigates early-stage reward hacking on auxiliary objectives (PickScore, HPSv2, etc.), thereby balancing the gradients across rewards and yielding an additional improvement in final GenEval2 performance over the unregularized baseline, without degrading end-of-training performance on any individual reward.

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 abstract and introduction clearly state the contributions, which are fully supported by the theoretical analysis in Section 3 and the experimental results in Section 4.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: The max-state versus sampled-state distinction, directional approximation, and Gaussian-policy scope are discussed in Appendix E.2 and Appendix F.3; the limited seed and video evaluations are stated in Appendix H.3.1 and Figure 3.

   Guidelines:

   - The answer [N/A] 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: Theorems 3.1 and 3.2 include full assumptions with complete proofs in Appendix C.

Guidelines:

- The answer [N/A] 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: All hyperparameters, architectures, and training details are provided in Section 4 and Appendix G.

   Guidelines:

   - The answer [N/A] 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: Code will be released after double-blind review.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: Training details including hyperparameters, data, and optimizer settings are specified in Section 4 and Appendix G.

   Guidelines:

   - The answer [N/A] 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: [No]

   Justification: Due to the prohibitive compute cost of diffusion-RL training, running each experiment multiple times is infeasible; see Appendix G for compute details.

   Guidelines:

   - The answer [N/A] 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: Compute resources are detailed in Appendix G.

   Guidelines:

   - The answer [N/A] 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 conforms with the NeurIPS Code of Ethics in every respect.

   Guidelines:

   - The answer [N/A] 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: Broader impacts are discussed in Appendix A.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper proposes a training algorithm and does not release high-risk models or data.

    Guidelines:

    - The answer [N/A] 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: All existing assets (models, datasets, and evaluation metrics) are properly cited with their original papers.

    Guidelines:

    - The answer [N/A] 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: [Yes]

    Justification: Released code is documented with instructions for reproducing the main experimental results.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: This work does not involve crowdsourcing or human subjects research.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: This work does not involve human subjects research.

    Guidelines:

    - The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not used as a component of the core methodology in this research.

    Guidelines:

    - The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.