Tencent Hunyuan

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

Bowen Ping[1,2,*] **Xiangxin Zhou**[2,*¶] **Penghui Qi**[3]
Minnan Luo[1,‡] **Liefeng Bo**[2] **Tianyu Pang**[2,‡]

[1]Xi'an Jiaotong University [2]Tencent Hunyuan [3]National University of Singapore

[*]Equal contribution [¶]Project Lead [‡]Corresponding author

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.

Date: June 8, 2026
Code: https://github.com/Tencent-Hunyuan/UniRL/tree/main/FlowDPPO

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 (Rafailov et al., 2023) and GRPO (Shao et al., 2024) have substantially improved alignment (Ouyang et al., 2022) and reasoning capabilities (Guo et al., 2025). Recently, these advances have been extended to image and video generation Liu et al. (2025); Wallace et al. (2024); Wang and Yu (2025); Xue et al. (2025a); Zheng et al. (2026), where flow matching models (Lipman et al., 2023; Liu et al., 2023) represent the dominant generative framework. Among them, Flow-GRPO (Liu et al., 2025) and DanceGRPO (Xue et al., 2025b)

Figure 1: Qualitative comparison on FLUX.1-dev (Black Forest Labs, 2024) with GenEval2 (Kamath et al., 2025) prompts. Flow-DPPO achieves competitive compositional accuracy with notably less image quality degradation compared to Flow-GRPO (Liu et al., 2025), Flow-CPS (Wang and Yu, 2025), and GRPO-Guard (Wang et al., 2025), reflecting their superior KL-proximal efficiency.

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) (Schulman et al., 2015), 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 (Schulman et al., 2017) later introduced ratio clipping as a computationally efficient first-order approximation to TRPO. However, as noted by Qi et al. (2026), each clipping decision is based on 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, leading to a systematic left shift in the ratio distribution, with its mean falling below one (Wang et al., 2025). We show that this bias is intrinsic to Gaussian policies: the standard PPO clipping range $[1-\epsilon, 1+\epsilon]$ therefore becomes effectively asymmetric, failing to adequately constrain over-optimization for positive-advantage samples while excessively clipping negative-advantage ones.

To mitigate this bias, GRPO-Guard (Wang et al., 2025) proposed normalizing the ratio distribution. While this re-centering 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 (Qi et al., 2026) 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 enforces the trust region while preserving the beneficial asymmetric structure of PPO: updates that move the policy *towards* 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 (Lipman et al., 2023; Liu et al., 2023) 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 $\boldsymbol{v} = \frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \dot{\alpha}_t \boldsymbol{x}_0 + \dot{\sigma}_t \boldsymbol{\epsilon}$ and the functinal $\dot{f}_t \coloneqq \mathrm{d}f_t/\mathrm{d}t$. The model $\boldsymbol{v}_\theta(\boldsymbol{x}_t, t)$ is then trained by minimizing the regression objective

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

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 (Karras et al., 2022; Lu et al., 2022; Song et al., 2021a). A notable special case is rectified flow (Liu et al., 2023), 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 (Shao et al., 2024) to flow matching models requires a sampler-induced stochastic policy at each denoising step. Flow-GRPO (Liu et al., 2025) constructs such a policy via an ODE-to-SDE conversion, which transforms the probability-flow ODE into an equivalent SDE with the same marginals (Albergo and Vanden-Eijnden, 2023; Albergo et al., 2024; Song et al., 2021b): $\mathrm{d}\boldsymbol{x}_t = \left[\boldsymbol{v}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}) + \frac{\sigma_t^2}{2t} \big(\boldsymbol{x}_t + (1-t) \boldsymbol{v}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}) \big) \right] \mathrm{d}t + \sigma_t \, \mathrm{d}\boldsymbol{w}$, 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) (Wang and Yu, 2025), which reduces the excessive noise injection in Flow-SDE and better preserves the interpolation structure of the scheduler. Let $\hat{\boldsymbol{x}}_0 = \boldsymbol{x}_t - t\,\hat{\boldsymbol{v}}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}), \hat{\boldsymbol{x}}_1 = \boldsymbol{x}_t + (1-t)\,\hat{\boldsymbol{v}}_\theta(\boldsymbol{x}_t, t, \boldsymbol{c})$ 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) (Black et al., 2024; Fan et al., 2023; Liu et al., 2025). 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}: $\max_\theta\ \mathbb{E}_{\boldsymbol{c}\sim p(\boldsymbol{c}),\,\tau\sim\pi_\theta}\Big[R(\boldsymbol{x}_0, \boldsymbol{c}) - \beta \sum_{k=1}^{K-1} D_{\text{KL}}\big(\pi_\theta(\cdot \mid \boldsymbol{s}_k) \,\|\, \pi_{\text{ref}}(\cdot \mid \boldsymbol{s}_k)\big)\Big]$, 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 (Liu et al., 2025) 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 (Schulman et al., 2017), the policy is optimized using the clipped surrogate objective

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

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 $r_k^i(\theta) = \frac{p_\theta(\boldsymbol{x}^i_{t_k-\Delta t} \mid \boldsymbol{x}^i_{t_k}, \boldsymbol{c})}{p_{\theta_{\text{old}}}(\boldsymbol{x}^i_{t_k-\Delta t} \mid \boldsymbol{x}^i_{t_k}, \boldsymbol{c})}$. 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 divergence mask, yielding a tighter and variance-free trust-region constraint.

3.1 Trust-Region Policy Optimization for Flow Matching Models

Inspired by Schulman et al. (2017); Qi et al. (2026), 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 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 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 $D_{\text{TV}}^{\max}(\pi_{\theta_{\text{old}}}\|\pi_\theta)=\max_{\boldsymbol{s}_k}D_{\text{TV}}\big(\pi_{\theta_{\text{old}}}(\cdot\mid\boldsymbol{s}_k)\|\pi_\theta(\cdot\mid\boldsymbol{s}_k)\big)$ is the maximum per-step Total Variation divergence, and $\xi=\max_{\boldsymbol{x}_0,\boldsymbol{c}}|R(\boldsymbol{x}_0,\boldsymbol{c})|$ is the maximum absolute reward.

Please refer to Appendix B for the detailed derivation; a tighter bound linear in K is given in Appendix B.3. This bound is structurally analogous to the policy improvement bound for LLMs derived in Qi et al. (2026). It provides a rigorous justification for trust-region methods in flow model fine-tuning: constraining the per-step divergence controls the penalty term and guarantees monotonic improvement. Similar to TRPO (Schulman et al., 2015), 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 (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:*

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

where Φ is the standard normal CDF. Constraining the TV divergence below a threshold is therefore equivalent to constraining $\|\boldsymbol{\mu}_{\theta_{\mathrm{old}}} - \boldsymbol{\mu}_\theta\|^2 \le \delta'$ for an appropriate δ', which is precisely the divergence measure that Flow-DPPO employs. Moreover, the Pinsker inequality $D_{\mathrm{TV}}(p\|q)^2 \le \frac{1}{2}D_{\mathrm{KL}}(p\|q)$ ensures that our KL-based constraint also upper-bounds the TV divergence: when the per-step $D_{\mathrm{KL}} \le \delta$, we have $D_{\mathrm{TV}}^{\max} \le \sqrt{\delta/2}$. In the Gaussian equal-covariance case, the converse also holds since KL and TV are both monotone functions of $\|\boldsymbol{\mu}_{\theta_{\mathrm{old}}} - \boldsymbol{\mu}_\theta\|/\sigma$. Thus, our method is theoretically justified from both the KL and TV perspectives. Unlike the LLM setting, where the discrete vocabulary requires approximate divergence computations (Qi et al., 2026), 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 (Liu et al., 2025), 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 $|r_t^i - 1| \le \epsilon$ 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,

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

so each individual $|r_t^i - 1|$ is merely a *single-sample Monte Carlo estimate* of $2\,D_{\mathrm{TV}}$. While the policy improvement bound (Theorem 2) calls for constraining $D_{\mathrm{TV}}^{\max}$, ratio clipping constrains this noisy per-sample surrogate instead. This issue was identified by Qi et al. (2026) 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:

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

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

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

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 D.

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 C for derivation):

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

For Flow-SDE (corresponding to Eq. (2)), $\sigma^2 = \sigma_t^2 \Delta t$, giving:

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

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

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

Remark 4. *In the LLM setting, DPPO (Qi et al., 2026) must approximate the true divergence via Binary or Top-K reductions of the vocabulary distribution, as computing exact TV or KL over $|\mathcal{V}| > 100K$ 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:

$$\mathcal{L}^{\text{Flow-DPPO}}(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{T}\sum_{t=0}^{T-1}\Big(M_t^i \cdot r_t^i(\theta)\cdot \hat{A}^i - \beta D_{\mathrm{KL}}(\pi_\theta\|\pi_{\mathrm{ref}})\Big)\right], \tag{17}$$

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 $D_t \equiv D_{\mathrm{KL}}\big(\pi_{\theta_{\text{old}}}(\cdot \mid \boldsymbol{x}_t^i) \,\|\, \pi_\theta(\cdot \mid \boldsymbol{x}_t^i)\big)$ and δ a 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 *towards* 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 E.

4 Experiments

Models and Baselines. We employ Stable Diffusion 3.5 Medium (Esser et al., 2024; Stability AI, 2024) (SD3.5), FLUX2-klein-base-9B (Black Forest Labs, 2026) (FLUX2-9B) and FLUX.1-dev (Black Forest Labs, 2024) as base models to cover diverse architectures and scales. We compare our method against four competitive baselines: Flow-GRPO (Liu et al., 2025), Flow-CPS (Wang and Yu, 2025), GRPO-Guard (Wang et al., 2025) and Diffusion-NFT (Zheng et al., 2026). 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 are deferred to Appendix F.

Metrics and Datasets. GenEval2 (Kamath et al., 2025) and PickScore (Kirstain et al., 2023) 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 (Kirstain et al., 2023), CLIP (Radford et al., 2021) score, and HPSv2 (Wu et al., 2023) during training. We report results for both single-reward optimization (GenEval2 only) and multi-reward training, where GDPO (Liu et al., 2026) aggregates advantages with equal reward weights.

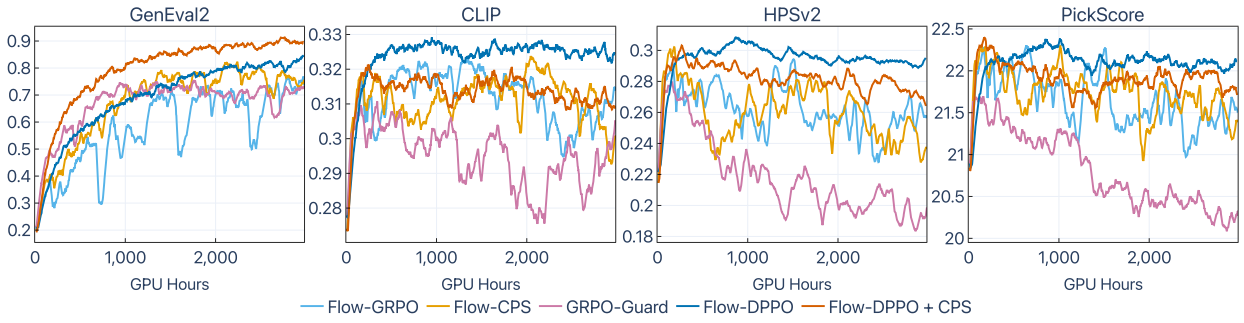

Figure 2: Training curves on FLUX2-9B for single-reward setting. Flow-DPPO variants achieve state-of-the-art performance and less catastrophic forgetting on out-of-domain rewards.

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 4 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**

4.1 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 4, where Flow-DPPO variants consistently outperform all baselines across most metrics on SD3.5, without sacrificing any individual objective.

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

Figure 3: Asymmetric masking ablation on SD3.5 with single-reward on GenEval2.

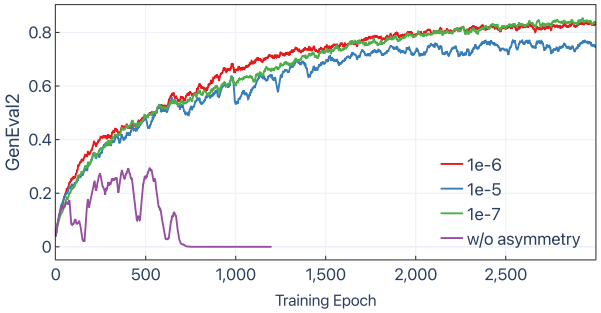

Table 2: KL divergence ($\times 10^{-3}$) between the RL fine-tuned model and the pre-trained reference. Lower is better. Full curves in Figure 12.

	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**

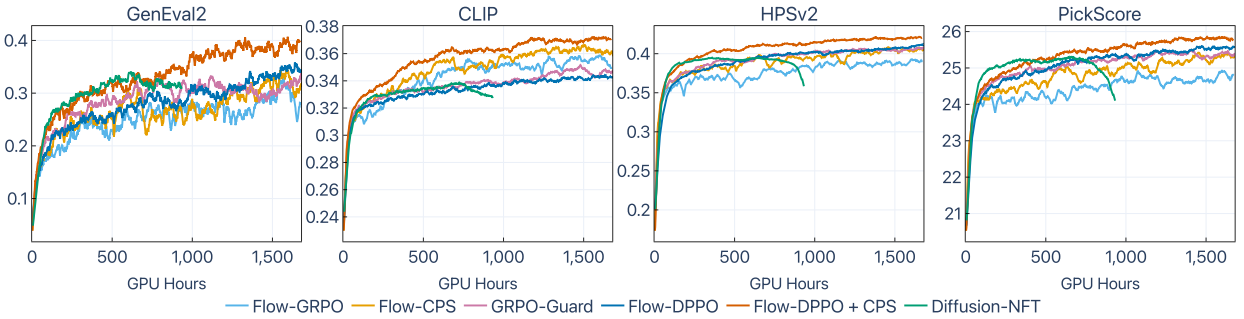

Figure 4: Training curves on SD3.5 for multi-reward setting. Flow-DPPO variants consistently outperform all baselines across all metrics.

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.

4.2 Analysis

Asymmetric Masking and Divergence Threshold. We investigate the impact of the divergence threshold and asymmetric masking in Flow-DPPO using SD3.5 with CPS sampling (Figure 3). Without asymmetric masking, the training process collapses as the trust-region regularization becomes ineffective; specifically, samples falling outside the trust region are largely ignored, preventing optimization progress. Conversely, asymmetric masking constrains these samples back within the trust region, thereby stabilizing the trajectory. Regarding the divergence threshold, a looser threshold (10^{-5}) results in diminished stability and suboptimal convergence. A tighter threshold (10^{-7}) initially slows down learning but fosters superior stability and slightly better final performance due to more rigorous trust-region enforcement.

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. Specifically, we vary two factors: (i) the number of groups per rollout, and (ii) the number of training epochs per rollout (inner loops). The latter determines the reuse frequency of each sample. For instance, two inner loops imply that each rollout batch is utilized for two consecutive gradient steps.

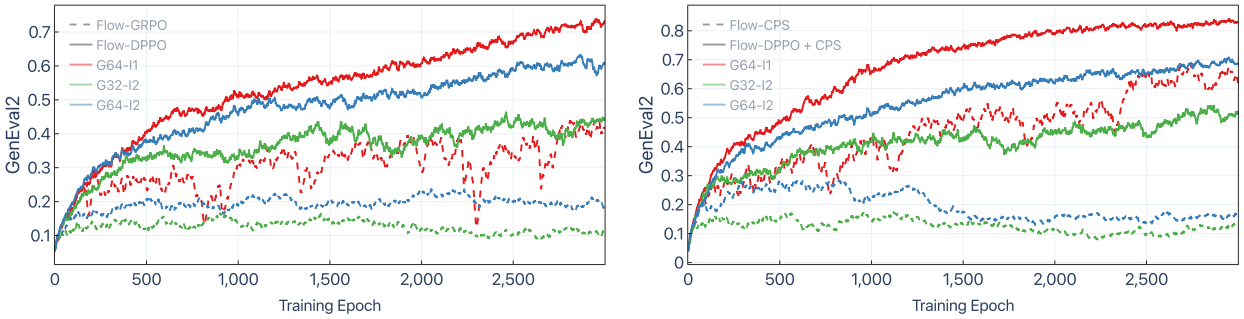

Figure 5: Multi-epoch training on SD3.5 (Left: Flow-SDE, Right: CPS). Flow-DPPO variants show consistent long-term gains under multi-epoch training (G64-I2 and G32-I2), while baselines plateau or even degrade.

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.

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.

5 Conclusion

We show ratio clipping in flow models is a noisy, biased 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 than baselines in terms of reward optimization and catastrophic forgetting. Furthermore, Flow-DPPO enables stable multi-epoch training where ratio clipping degrades, offering a promising direction for scenarios with expensive rollouts, such as long-video generation.

References

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

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.

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.

Black Forest Labs. Flux.1: Announcing black forest labs. https://blackforestlabs.ai/announcing-black-forest-labs/, 2024.

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.

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.

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.

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.

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

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

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

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.

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

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*, 2021b.

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.

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.

Feng Wang and Zihao Yu. Coefficients-preserving sampling for reinforcement learning with flow matching. *arXiv preprint arXiv:2509.05952*, 2025.

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.

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.

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*, 2025a.

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*, 2025b.

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.

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

Appendix A. The Flow-DPPO Algorithm

We summarize the complete Flow-DPPO training procedure. The algorithm adopts the CPS sampling framework (Wang and Yu, 2025) for trajectory generation, uses group-relative advantage estimation, and applies the divergence-based mask during policy optimization.

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
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(Eq. (2)) or CPS (Eq. (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 divergence $D_t = \|\boldsymbol{\mu}_{\theta_{\text{old}}}(\boldsymbol{x}_t^i, t) - \boldsymbol{\mu}_\theta(\boldsymbol{x}_t^i, t)\|^2$
    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

Computational overhead. The divergence computation requires one additional forward pass of the velocity network to obtain $\boldsymbol{\mu}_\theta(\boldsymbol{x}_t^i, t)$ at training time. However, this forward pass is already required for computing the log ratio, so the divergence $D_t = \|\boldsymbol{\mu}_{\theta_{\text{old}}} - \boldsymbol{\mu}_\theta\|^2$ comes at *zero additional cost*: it is simply the squared norm of a difference that is already computed.

Appendix B. Policy Improvement Bound for Flow Models

We adapt the classical policy improvement theory (Kakade and Langford, 2002; Schulman et al., 2015) to the finite-horizon, undiscounted setting of flow model denoising, following the approach of Qi et al. (2026) for the LLM regime. We use the MDP notation introduced in Section 2.1: $K-1$ decision steps indexed by $k \in \{1, \ldots, K-1\}$, states $\boldsymbol{s}_k = (\boldsymbol{c}, t_k, \boldsymbol{x}_{t_k})$, actions $\boldsymbol{a}_k = \boldsymbol{x}_{t_{k+1}}$, and terminal reward $R(\boldsymbol{x}_0, \boldsymbol{c})$.

B.1 Proof of Performance Difference Identity

Proof [Proof of Theorem 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 $\tau = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_{K-1})$ (we omit the deterministic transition structure for notational clarity).

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 $\prod_{k=1}^{N} a_k - \prod_{k=1}^{N} b_k = \sum_{k=1}^{N} \big(\prod_{j=1}^{k-1} b_j\big)(a_k - b_k)\big(\prod_{j=k+1}^{N} a_j\big)$:

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

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:

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

This completes the proof. ■

B.2 Proof of Policy Improvement Bound

Lemma 5 (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:*

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

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

Proof Let $P(\tau_{>k}) = \pi_{\theta_{\text{old}},>k}(\tau_{>k} \mid \boldsymbol{s}_{k+1})$ 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:

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

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:

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

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:

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

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

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

■

Proof [Proof of Theorem 2] From Theorem 1, we start with the exact performance difference identity:

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

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

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

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

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

Applying Theorem 5 and bounding each term by $D_{\text{TV}}^{\max}$:

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

Substituting back into Eq. (19):

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

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

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

Substituting into the performance difference identity yields the desired bound:

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

This completes the proof. ■

B.3 A Tighter Policy Improvement Bound

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

Starting from the intermediate step in Eq. (19), the inner expectation is $2D_{\text{TV}}(\pi_{\theta_{\text{old}},>k}(\cdot \mid \boldsymbol{s}_{k+1}) \| \pi_{\theta,>k}(\cdot \mid \boldsymbol{s}_{k+1}))$. Instead of applying Theorem 5, we directly use the universal bound $D_{\text{TV}} \le 1$:

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

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

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

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

B.4 Connection to Gaussian Per-Step Divergence

For the Gaussian policies in Eq. (4), $\pi_{\theta_{\text{old}}}(\cdot \mid \boldsymbol{s}_k) = \mathcal{N}(\boldsymbol{\mu}_{\theta_{\text{old}}}, \sigma^2(t_k)\mathbf{I})$ 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:

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

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

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

This formally establishes that the Flow-DPPO mask, which blocks updates when $\|\boldsymbol{\mu}_{\theta_{\text{old}}} - \boldsymbol{\mu}_\theta\|^2 > \delta$, implements a trust-region constraint equivalent (up to a monotone rescaling) to constraining the per-step TV divergence. The policy improvement bound (Theorem 2) thus provides a rigorous theoretical guarantee for Flow-DPPO: by enforcing a per-step divergence threshold, the penalty term remains controlled, ensuring monotonic policy improvement.

Appendix C. 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.

C.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:

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

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

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

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

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

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

C.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:

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

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

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

C.4 Application to CPS

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

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

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

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

In previous work (Wang and Yu, 2025), the $2\sigma_{\text{CPS}}^2$ normalization is dropped for numerical stability, reducing the divergence to $D(\pi_{\theta_{\text{old}}} \| \pi_\theta) = \|\boldsymbol{\mu}_{\theta_{\text{old}}}^{\text{CPS}} - \boldsymbol{\mu}_\theta^{\text{CPS}}\|^2$. We instead retain the full normalization in Eq. (22): because $\sigma_{\text{CPS}}^2 \propto (t-\Delta t)^2$ shrinks at later denoising steps, the σ_{CPS}^{-2} factor amplifies the divergence where small velocity changes most affect the output, yielding a tighter constraint that prevents distribution collapse.

Appendix D. Ratio Variance Analysis

We provide a detailed analysis of the variance of the log-ratio in flow models.

From Eq. (13), $\log r_t^i = \boldsymbol{\epsilon}^\top \boldsymbol{d}/\sigma - \|\boldsymbol{d}\|^2/(2\sigma^2)$, where $\boldsymbol{d} = \boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\theta_{\text{old}}}$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. It follows that:

$$\mathbb{E}[\log r_t^i] = -\frac{\|\boldsymbol{d}\|^2}{2\sigma^2} = -D_{\text{KL}}(\pi_{\theta_{\text{old}}} \| \pi_\theta), \qquad \text{Var}[\log r_t^i] = \frac{\|\boldsymbol{d}\|^2}{\sigma^2} = 2\,D_{\text{KL}}(\pi_{\theta_{\text{old}}} \| \pi_\theta).$$

Thus $\text{std}[\log r_t^i] = \sqrt{2\,D_{\text{KL}}}$. When the KL is moderate (e.g., $D_{\text{KL}} = 0.5$), the standard deviation of the log-ratio is 1.0, meaning that individual log-ratio samples fluctuate by ± 1 around the mean of -0.5. In terms of the ratio itself, this corresponds to roughly a $3\times$ multiplicative spread.

Implication for clipping. With a typical clip parameter $\epsilon = 0.2$ (i.e., clip range $[0.8, 1.2]$), the log-clip range is $[\log 0.8, \log 1.2] \approx [-0.22, 0.18]$. Comparing this narrow range with the log-ratio standard deviation of $\sqrt{2\,D_{\text{KL}}}$, we see that even for modest KL values, a significant fraction of samples will be clipped purely due to noise, not because the true divergence is excessive. This provides rigorous justification for replacing ratio-based clipping with direct divergence measurement.

Appendix E. 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 $\pi_{\theta_{\text{old}}}$. 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:

$$M_t^i = 0 \quad \Longleftrightarrow \quad \operatorname{sgn}\big(\hat{A}^i \cdot (r_t^i - 1)\big) > 0 \quad \wedge \quad D_t > \delta. \tag{23}$$

While this design is effective in practice, the directional indicator $\operatorname{sgn}\big(\hat{A}^i(r_t^i - 1)\big)$ 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 $D_t = D_{\text{KL}}(\pi_{\theta_{\text{old}}} \| \pi_\theta)$, 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 (§E.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 $\operatorname{sgn}\big(\hat{A} \cdot (\log r_t - D_t)\big)$, 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 (§E.2) that directly forecasts whether the post-update divergence will exceed δ.

E.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 η, the updated mean is $\boldsymbol{\mu}_{\text{new}} = \boldsymbol{\mu} + \eta \cdot \nabla_{\boldsymbol{\mu}} L$. Let $\boldsymbol{g} = \sigma\boldsymbol{\epsilon} - \boldsymbol{d}$. The predicted post-update divergence is:

$$\begin{aligned} D_t^{\text{new}} &= \frac{\|\boldsymbol{\mu}_{\text{new}} - \boldsymbol{\mu}_{\text{old}}\|^2}{2\sigma^2} = \frac{1}{2\sigma^2}\left\|\boldsymbol{d} + \frac{\eta\,\hat{A}\,r_t}{\sigma^2}\,\boldsymbol{g}\right\|^2 \\ &= D_t + \frac{\eta\,\hat{A}\,r_t}{\sigma^4}\,\boldsymbol{g}^\top\boldsymbol{d} + \frac{\eta^2\hat{A}^2 r_t^2}{2\sigma^6}\|\boldsymbol{g}\|^2. \end{aligned} \tag{24}$$

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 $(\eta\, \hat{A}\, r_t/\sigma^2)(\log r_t - D_t)$. The three terms in Eq. (24) 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 η 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 $\eta > 0$, this sign equals:

$$\operatorname{sgn}\Big(\hat{A} \cdot \big(\log r_t - D_t\big)\Big). \tag{25}$$

When this is positive, the gradient step increases D_t; when negative, it decreases D_t. 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. (23). 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*:

$$M_t^{(1)} = \begin{cases} 0, & \text{if } \operatorname{sgn}\Big(\hat{A} \cdot \big(\log r_t - D_t\big)\Big) > 0 \ \wedge\ D_t > \delta, \\ 1, & \text{otherwise.} \end{cases} \tag{26}$$

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.

E.2 The Predictive Mask

Based on Eq. (24), we define the *(full) predictive mask* that blocks updates whenever the predicted post-update divergence would exceed δ:

$$M_t^{\text{pred}} = \begin{cases} 0, & \text{if } D_t^{\text{new}} > \delta, \\ 1, & \text{otherwise.} \end{cases} \tag{27}$$

Comparison with the first-order mask. The first-order mask (Eq. (26)) only considers the direction of divergence change and still relies on the separate threshold condition $D_t > \delta$. The full predictive mask unifies both into a single inequality: whether the gradient increases or decreases divergence is automatically encoded in the predicted value D_t^{new}, and the threshold comparison is applied to the predicted (rather than current) divergence. This has two consequences. First, when

$D_t \ll \delta$, even a divergence-increasing step may be permitted if the predicted D_t^{new} remains below δ. Second, when D_t is close to δ, the second-order term $\eta^2\|\boldsymbol{g}\|^2$ may push D_t^{new} above δ even when the first-order direction is "safe" (i.e., the first-order mask would not fire), correctly blocking large gradient steps near the trust-region boundary.

Recovery of the existing mask. In the limit $\eta \to 0$, the second-order term vanishes and $D_t^{\text{new}} > \delta$ reduces to requiring that the first-order direction is positive and $D_t > \delta$. Combined with the small-divergence approximation ($D_t \approx 0$), this exactly recovers the current Flow-DPPO mask (Eq. (23)).

E.3 Discussion on Mask Variants

Hierarchy of masks. The three masks form a natural hierarchy of increasing fidelity:

$$\underbrace{\operatorname{sgn}\big(\hat{A}(r_t-1)\big)}_{\text{current (Eq. (23))}} \subset \underbrace{\operatorname{sgn}\big(\hat{A}(\log r_t - D_t)\big)}_{\text{first-order (Eq. (26))}} \subset \underbrace{D_t^{\text{new}} > \delta}_{\text{full predictive (Eq. (27))}} .$$

The current mask is the cheapest (no additional computation) and suffices when the trust region keeps D_t small throughout training. The first-order mask refines the directional decision with zero additional hyperparameters. The full predictive mask additionally requires an effective learning rate estimate but provides quantitative divergence prediction.

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.

Appendix F. Experimental Details

F.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.

F.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.

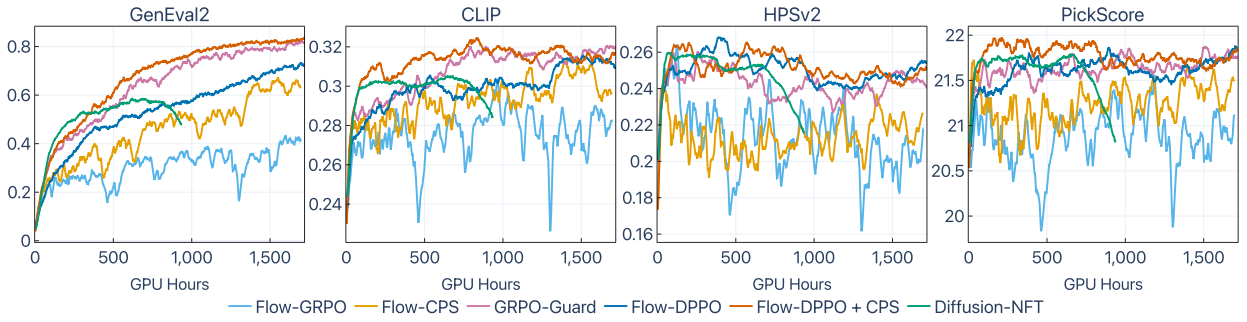

Figure 7: Training curves on SD3.5 for single-reward setting, including Diffusion-NFT (Zheng et al., 2026) as an additional baseline. Flow-DPPO variants achieve state-of-the-art performance and less catastrophic forgetting on out-of-domain rewards, consistent with the main results.

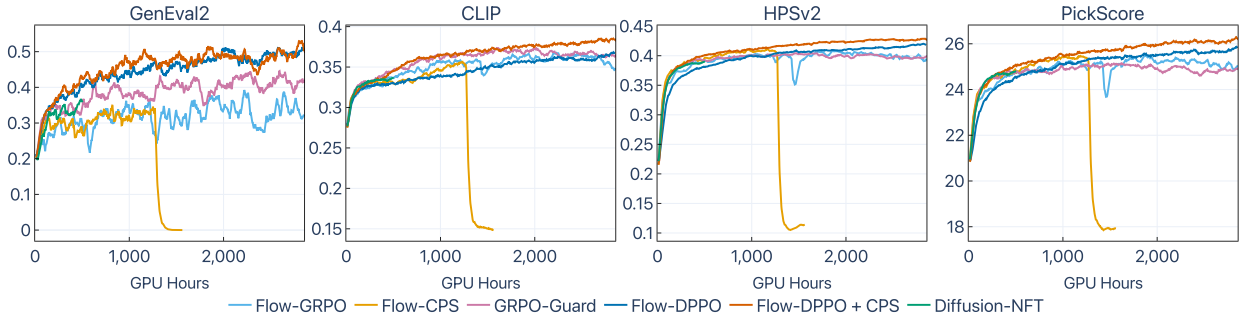

Figure 8: Training curves on FLUX2-9B for multi-reward setting (GPU hours). Flow-DPPO variants consistently outperform the baselines across all metrics, with a notable improvement on the GenEval2 reward.

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 stragegy proposed in MixGRPO (Li et al., 2025) on all baselines and proposed methods 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.

Appendix G. Additional Experimental Results

G.1 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 4 in the main body). We also provide the FLUX2-9B multi-reward training curves in Figure 8 and FLUX.1-dev in Figure 9.

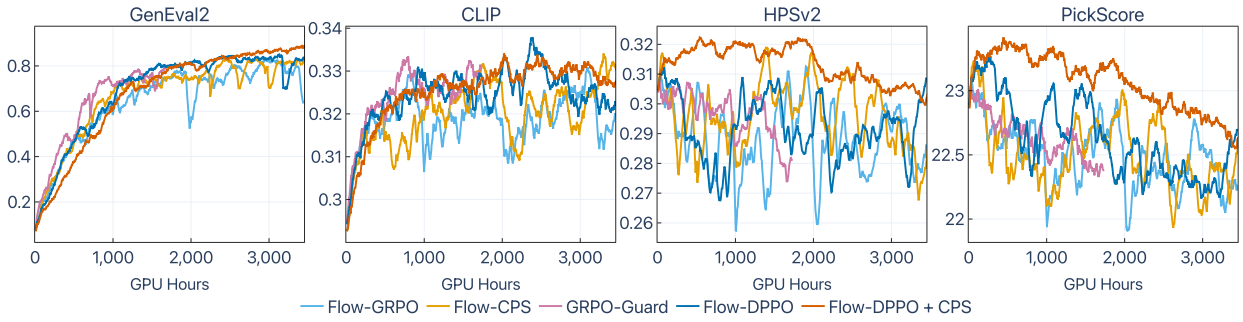

Figure 9: Training curves on FLUX.1-dev for single-reward setting.

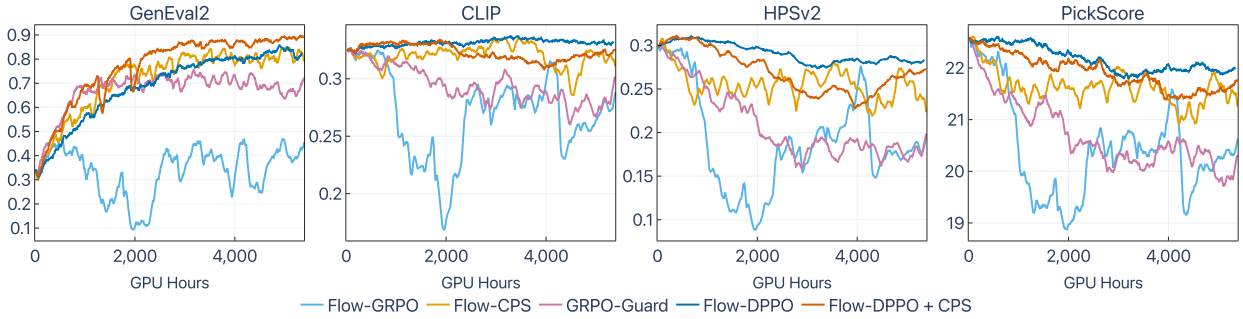

Figure 10: Training curves on FLUX2-9B with CFG scale 4.0. Flow-DPPO variants remain robust under CFG, achieving strong performance with less catastrophic forgetting.

We additionally provide training curves on FLUX.1-dev in Figure 9.

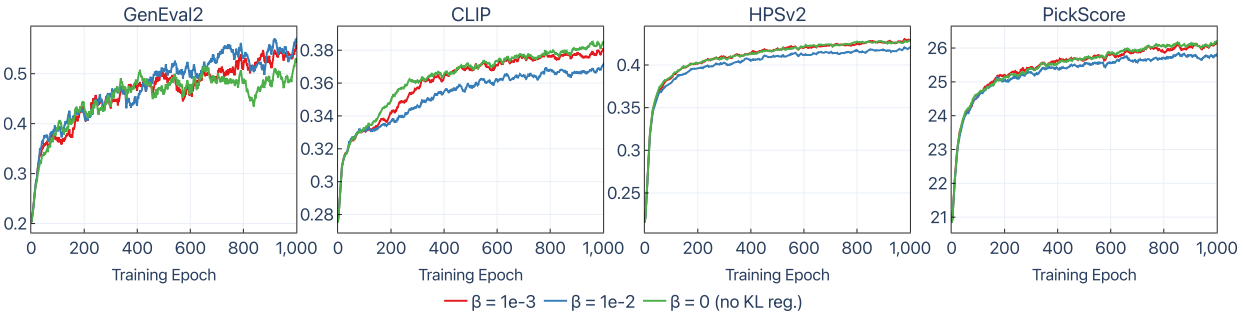

Figure 11: Training reward curves under three $D_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{ref}})$ regularization strengths (β) on FLUX2-klein-base-9B (multi-reward GDPO, CPS schedule). A moderate β=10^{-3} suppresses early reward hacking on PickScore and HPSv2, balancing cross-reward gradients and boosting final GenEval2 performance without hurting end-of-training performance on any individual reward.

G.2 KL Divergence Curves

Figure 12 visualises 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.

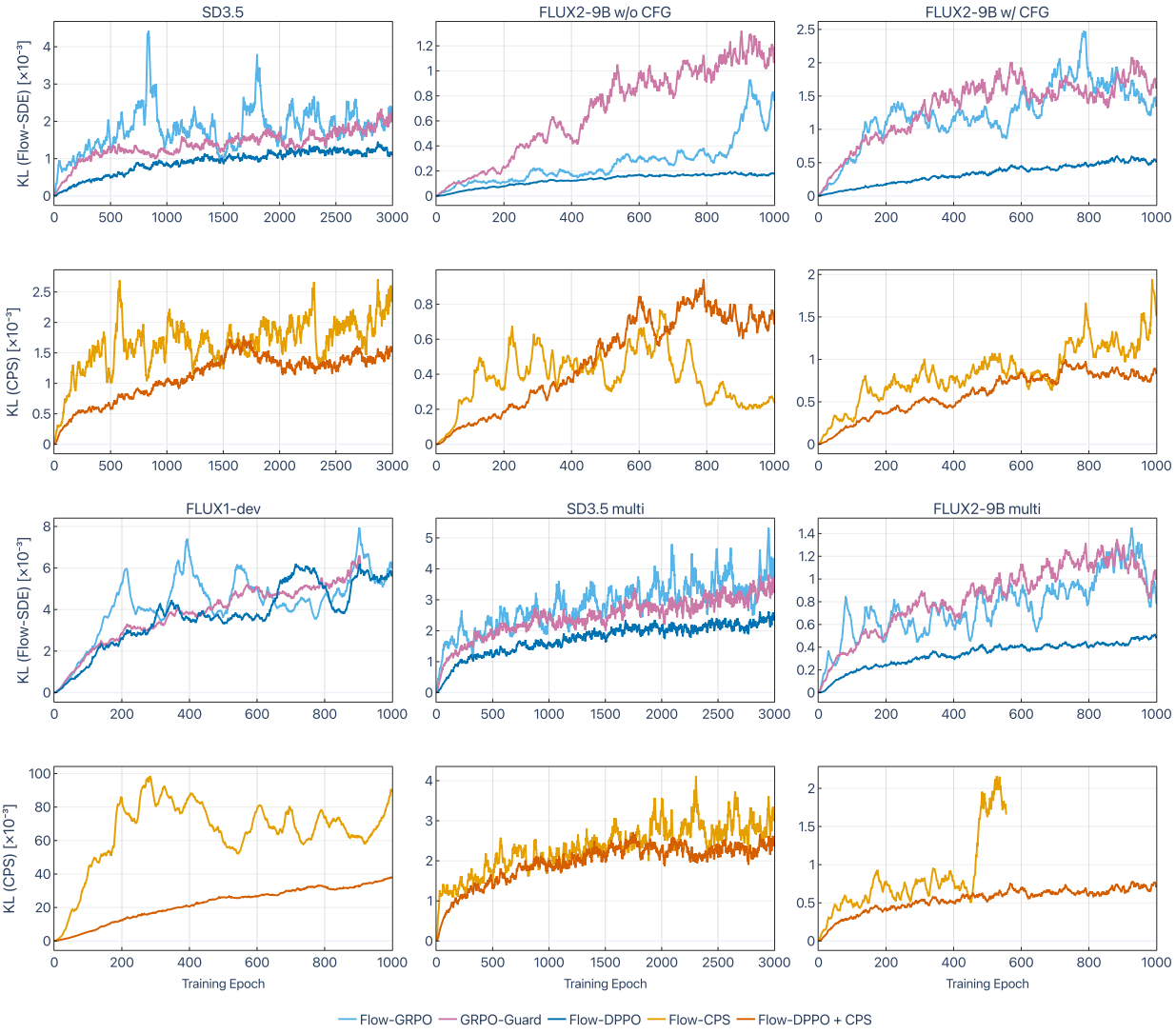

Figure 12: KL-divergence between the current and reference (pre-trained) model during training, across six training settings (columns: four single-reward — SD3.5, FLUX2-9B w/o CFG, FLUX2-9B w/ CFG, FLUX.1-dev; two multi-reward — SD3.5 multi, FLUX2-9B multi) and two SDE schedules (rows: Flow-SDE, CPS). 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.

G.3 Ablation Studies

G.3.1 Classifier-Free Guidance

Previous works found that CFG heavily affects the training convergence and performance (Zheng et al., 2026). Here, we study the effect of CFG on the training of Flow-DPPO on FLUX2-9B, as shown

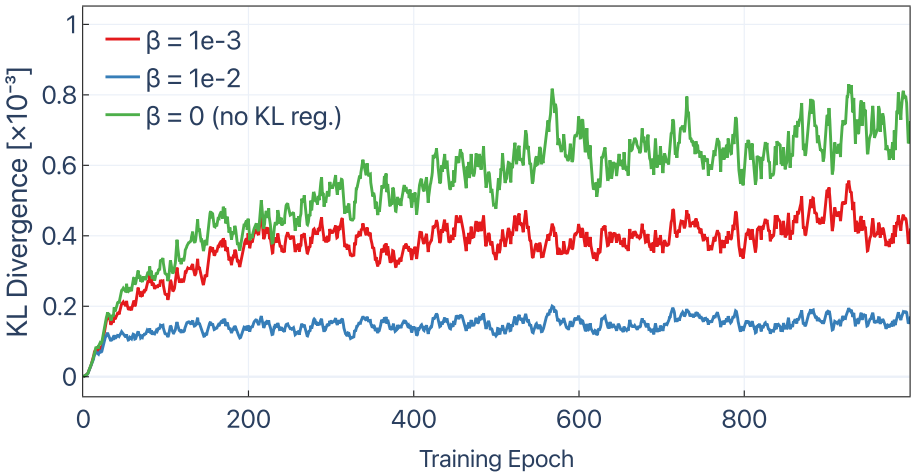

Figure 13: $D_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{ref}})$ during training for different β settings.

Table 3: End-of-training Soft $\mathrm{TIFA}_{\mathrm{GM}}$ on GenEval2 (%) across six training configurations (columns) and five RL algorithms (rows). The six columns correspond, left-to-right, to Figs. 2, 8, 10, 7, 4, 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	<u>89.0</u>	74.8	44.6	<u>91.2</u>
GRPO-Guard	82.8	49.0	78.8	**85.8**	47.8	87.6
Diffusion-NFT	–	47.3	–	64.5	42.5	–
Flow-DPPO	<u>85.1</u>	**57.7**	87.4	78.9	<u>48.1</u>	90.7
Flow-DPPO + CPS	**92.6**	<u>55.2</u>	**91.0**	<u>84.1</u>	**51.6**	**91.6**

in Figure 10, 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.

G.3.2 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 11 shows the training reward curves and Figure 13 shows the KL divergence from the pretrained model. A moderate regularization strength (β=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.

G.4 Quantitative Summary on GenEval2

To complement the per-setting training-curve figures above, Tables 4 and 3 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, contextualising 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 3 then expands the per-method Soft TIFA_{GM} comparison to all five training settings reported in this paper.

Table 4: GenEval2 [Soft TIFA_{GM}, defined in (Kamath et al., 2025)] 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