REWARDFLOW: LEARNING TO ALIGN FLOW MODELS WITH REWARD DISTRIBUTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-training alignment is important for flow-based text-to-image generation, but existing GRPO-style methods usually optimize scalar rewards and tend to concentrate the policy on a few reward-favored samples. This limits diversity, weakens generalization to unseen preference metrics, and becomes unstable under multi-reward optimization. In this paper, we propose **RewardFlow**, **a reward distribution matching framework** for post-training flow matching models. Instead of directly maximizing scalar rewards, RewardFlow constructs a reward-induced distribution over sampled denoising trajectories and aligns the flow policy with this distribution, allowing multiple high-quality generations to receive positive optimization signals. To stabilize image flow optimization, we use a warm-started learnable prompt-conditioned partition function for reward calibration and a step-wise KL constraint to preserve the pretrained flow prior. Experiments on SD3.5-M and FLUX.1-dev under both single-reward and multi-reward settings show that RewardFlow achieves stronger preference alignment, better generalization, and more diverse generations. Notably, under multi-reward training on SD3.5-M, RewardFlow improves HPS-v2.1 from 0.326 to 0.362 and the unseen Aesthetic Score from 6.226 to 6.470 over Flow-GRPO. Our code will be released on GitHub.

1 INTRODUCTION

As pretrained Flow Matching models Lipman et al. (2022); Liu et al. (2022; 2026); Albergo et al. (2025); Gao et al. (2024) become increasingly mature, post-training alignment has emerged as a central challenge for generative vision models. The key bottleneck is aligning model outputs with human preferences and aesthetic standards Ouyang et al. (2022) while maintaining high generation quality. Recent methods, particularly DanceGRPO Xue et al. (2025) and Flow-GRPO Liu et al. (2025b), transfer Group Relative Policy Optimization (GRPO) Shao et al. (2024) from large language models to flow-based image generation. By rewarding high-scoring samples and suppressing low-scoring ones within each prompt group, they align the generation policy with a given reward model and have rapidly become a mainstream paradigm for post-training flow-based generative models.

However, methods such as Flow-GRPO Liu et al. (2025b) still suffer from a fundamental limitation: they simplify the optimization objective to scalar reward maximization. These approaches reduce each image to a scalar reward and tend to overfit the single generation directions favored by the training reward, making it difficult to preserve multiple high-quality alternatives. This issue becomes particularly severe in text-to-image generation. A single prompt can admit multiple valid outputs across semantics, composition, and style, but a scalar reward cannot represent how probability should be distributed among them. This leads to three observable issues in Fig. 1: **(1) Reduced diversity.** The left panel shows that outputs of Flow-GRPO under the same prompt become highly similar, with repeated compositions, color palettes, and semantic layouts. **(2) Limited generalization.** Flow-GRPO lags behind RewardFlow on both the optimized HPS-v2.1 reward Wu et al. (2023) and unseen reward models as shown in the top-right panel. **(3) Conflicts in multi-reward optimization.** Optimizing a single scalar reward cannot fully capture the distinct optimization directions of multiple rewards, which leads to conflicts among different preference criteria during training. Consequently, even after 600 epochs, Flow-GRPO drops from 0.360 in the single-reward setting to 0.326 in the multi-reward setting on HPS-v2.1, revealing a clear optimization gap caused by multi-reward conflicts.

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

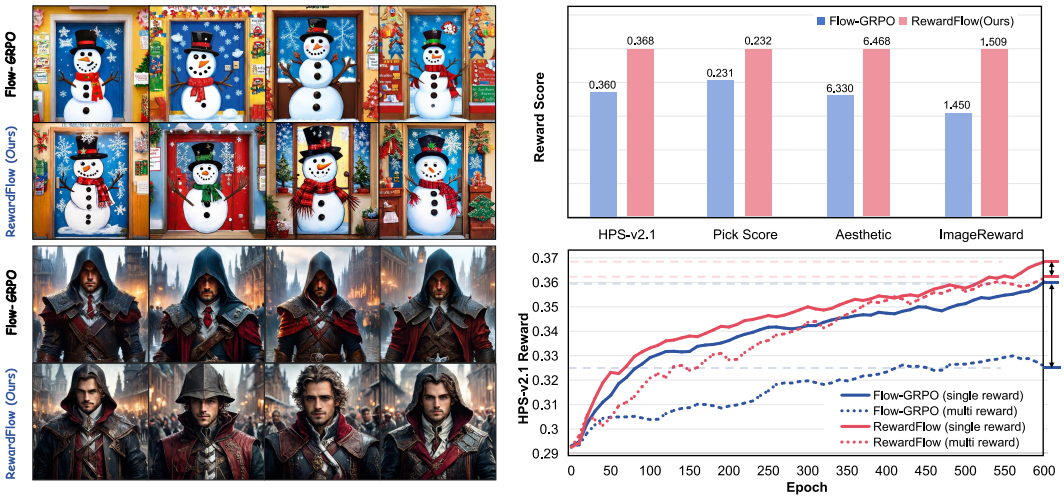

Figure 1: **RewardFlow improves diversity, generalization, and multi-reward consistency over Flow-GRPO. Left:** Flow-GRPO generates similar samples for the same prompt, while RewardFlow preserves diverse high-quality outputs. **Top right:** Under single-reward training, RewardFlow improves both the optimized HPS-v2.1 reward and unseen reward metrics, showing stronger generalization. **Bottom right:** Flow-GRPO drops under multi-reward training despite including HPS-v2.1, whereas RewardFlow remains more consistent across reward settings.

In this paper, we propose **RewardFlow**, a new post-training framework for flow-based image generation that shifts from scalar reward maximization to reward distributions. Our key insight is to align the flow policy with a prompt-wise reward-induced distribution, so that probability is allocated across multiple preferred samples rather than concentrated on only the highest-scoring ones. Specifically, given a prompt, RewardFlow samples a group of generation trajectories, evaluates each trajectory using reward models, converts the resulting scores into a probability distribution via a learnable prompt-conditioned normalization function, and aligns the flow-policy distribution with this reward distribution. Compared with original optimization objectives, RewardFlow avoids driving the policy toward only one reward-favored generation direction and helps preserve multiple high-quality alternatives. However, directly matching the reward distribution can pull the policy away from the pretrained flow prior. To avoid this, we impose a step-wise KL constraint on denoising transitions. We further warm-start the partition head to stabilize early training without changing the reward-induced target distribution. These designs improve diversity and generalization while preserving generation capability, with no additional training cost over Flow-GRPO.

To evaluate the effectiveness of RewardFlow, we evaluate under single-reward and multi-reward training on SD3.5-M Esser et al. (2024) and FLUX.1-dev Labs (2024). Results show that RewardFlow outperforms Flow-GRPO with stronger and more stable preference alignment. Under single-reward training, it achieves the best HPS-v2.1 scores on both backbones and improves unseen reward metrics. In the multi-reward setting, RewardFlow improves HPS-v2.1 from **0.326** to **0.362** on SD3.5-M, and also raises the unseen Aesthetic Score Schuhmann from **6.226** to **6.470**, with consistent gains on Pick Score and ImageReward. These results demonstrate that our method provides better stability, generalization, and robustness.

Our contributions are summarized as follows:

- We introduce RewardFlow, a new post-training framework for flow-based image generation, which transforms the optimization objective from scalar reward maximization to matching prompt-wise reward distributions.
- We propose a warm-started learnable prompt-conditioned normalization function to convert scalar rewards over sampled trajectories into a reward distribution, together with step-wise KL regularization to keep the policy close to the pretrained flow prior.
- Extensive experiments show that RewardFlow achieves stronger preference alignment under both single-reward and multi-reward training settings. Notably, under multi-reward training

on SD3.5-M, RewardFlow improves HPS-v2.1 from 0.326 to 0.362 and increases the unseen Aesthetic Score from 6.226 to 6.470.

2 Related Work

Reinforcement Learning for Flow-based Generative Models. Recent works extend reinforcement learning to flow-based visual generation for task-specific rewards and human preferences. Flow-GRPO applies online policy-gradient RL to flow matching via ODE-to-SDE conversion and denoising reduction Liu et al. (2025b), while DanceGRPO generalizes GRPO-style optimization to diffusion, rectified flows, and image/video generation Xue et al. (2025). Follow-up works improve this paradigm through mixed ODE–SDE sampling Li et al. (2025a), credit assignment Ding & Ye (2025); He et al. (2025); Lyu et al. (2025); Zhou et al. (2025); Li et al. (2025b); Luo et al. (2025); Lee & Ye (2025), and reward design Deng et al. (2026); Chen et al. (2025); Zhong et al. (2026); Shao et al.. However, these methods primarily optimize scalar rewards or group-relative advantages, focusing on reward maximization rather than probability allocation across preferred generations. RewardFlow instead converts group-level preference scores into a prompt-wise reward-induced distribution and matches the flow policy to it.

Reward Distribution Matching and Generative Flow Networks. Generative Flow Networks (GFlowNets) learn stochastic policies that sample terminal objects proportional to their rewards, providing a distributional alternative to reward maximization Bengio et al. (2023). Trajectory Balance further enables trajectory-level optimization for long-horizon credit assignment Malkin et al. (2022), while FlowRL extends reward-distribution matching to LLM reasoning Zhu et al. (2025). These methods mainly target discrete objects or token-level reasoning, whereas RewardFlow addresses continuous flow-based image generation, where preference alignment operates over denoising trajectories of latent states and generation actions. RewardFlow thus constructs reward-induced distributions over image trajectories, uses a learnable prompt-conditioned function for probability calibration, and applies step-wise KL regularization to preserve the pretrained flow prior.

3 Preliminaries: Flow-GRPO with ODE-to-SDE Sampling

Flow Matching models learn a time-dependent velocity field $v_\theta(x_t, t, c)$ that transports an initial noise distribution to the data distribution conditioned on a text prompt c. The standard sampling process follows a deterministic probability flow ODE:

$$\frac{dx_t}{dt} = v_\theta(x_t, t, c), \qquad t : 1 \to 0. \tag{1}$$

Starting from an initial noise latent $x_N \sim \mathcal{N}(0, I)$, the model generates a denoising trajectory $\tau = (x_N, x_{N-1}, \ldots, x_0)$, and the final image is decoded as $y = \mathcal{D}(x_0)$.

To enable reinforcement learning over the generation process, this iterative denoising procedure can be viewed as a Markov Decision Process (MDP). At step n, the state is defined as $s_n = (c, n, x_n)$, and the action is the next latent $a_n = x_{n-1}$. The policy is therefore the conditional distribution $\pi_\theta(a_n \mid s_n) = p_\theta(x_{n-1} \mid x_n, c)$. The transition is deterministic given the action, and the next state becomes $s_{n-1} = (c, n-1, x_{n-1})$.

However, purely deterministic sampling limits exploration and makes policy optimization unstable under reward feedback. To address this, Flow-GRPO adopts an ODE-to-SDE reformulation, introducing controlled stochasticity at each denoising step:

$$K_\theta^n(x_{n-1} \mid x_n, c) = \mathcal{N}\left(x_{n-1}; \mu_\theta^n(x_n, c), \sigma_n^2 I\right), \tag{2}$$

where $\mu_\theta^n(x_n, c)$ is determined by flow prediction and scheduler update, and σ_n controls noise level.

Under this formulation, Flow-GRPO treats each denoising step as a policy rollout. Flow-GRPO samples multiple trajectories per prompt, evaluates them with a reward model, and updates the policy using relative advantages. This formulation provides the foundation for extending scalar reward optimization to distributional matching over trajectories.

4 REWARDFLOW

4.1 REWARD-INDUCED DISTRIBUTION OVER FLOW TRAJECTORIES

From Scalar Rewards to Reward-Induced Distributions. Following Section 3, let $R(c,\tau) = r(D(x_0), c)$ denote the terminal reward. Existing RL-based post-training methods typically optimize

$$\max_{\theta} \mathbb{E}_{x_N\sim\mathcal{N}(0,I),\,\tau\sim\pi_\theta(\cdot|x_N,c)}\left[R(c,\tau)\right], \tag{3}$$

where $\pi_\theta(\tau \mid x_N, c)$ denotes the trajectory distribution induced by the current flow policy. This objective can concentrate updates on a few high-reward trajectories. As illustrated in Fig. 2, the highest-reward trajectory dominates the optimization signal, causing the updated flow field to concentrate around a narrow high-reward region. As a result, these kinds of methods lead to reduced diversity, weaker generalization, and conflicts when multiple rewards prefer different visual attributes.

RewardFlow addresses this limitation by converting scalar rewards into a reward-induced distribution, as shown in Fig. 2. To turn this into a probability weight, we use an exponential transformation:

$$\omega_\beta(\tau, c) = \exp(\beta R(c,\tau)). \tag{4}$$

This transformation maps scalar rewards to positive weights and turns reward differences into probability ratios $\frac{\omega_\beta(\tau_1,c)}{\omega_\beta(\tau_2,c)} = \exp\left(\beta\left[R(c,\tau_1) - R(c,\tau_2)\right]\right)$. Hence, higher-reward trajectories receive larger weights, while β controls the concentration on high-reward regions.

However, the reward weight alone does not define a complete trajectory distribution. It only specifies the relative preference between trajectories, but not the reference measure on which these weights are applied. Since imperfect reward models can favor unnatural or over-optimized samples, we reweight the frozen pretrained distribution $\pi_{\mathrm{ref}}(\tau \mid x_N, c)$ within its supported trajectory region:

$$\tilde{q}_\beta(\tau \mid x_N, c) = \pi_{\mathrm{ref}}(\tau \mid x_N, c)\exp(\beta R(c,\tau)). \tag{5}$$

Normalizing gives the reward-induced trajectory distribution:

$$q^\star_\beta(\tau \mid x_N, c) = \frac{\pi_{\mathrm{ref}}(\tau \mid x_N, c)\exp(\beta R(c,\tau))}{Z_\beta(x_N, c)},\ Z_\beta(x_N, c) = \int \pi_{\mathrm{ref}}(\tau \mid x_N, c)\exp(\beta R(c,\tau))d\tau. \tag{6}$$

where $Z_\beta(x_N, c)$ is the trajectory-level partition function. It normalizes the reward-reweighted trajectory density for a prompt and initial noise, making the target a valid probability distribution. We match the current flow policy to this distribution by minimizing:

$$\min_{\theta} D_{\mathrm{KL}}\left(\pi_\theta(\cdot \mid x_N, c) \| q^\star_\beta(\cdot \mid x_N, c)\right). \tag{7}$$

Expanding this KL gives:

$$\begin{aligned} D_{\mathrm{KL}} &= \mathbb{E}_{\tau\sim\pi_\theta}\left[\log \pi_\theta(\tau \mid x_N, c) - \log q^\star_\beta(\tau \mid x_N, c)\right] \\ &= \mathbb{E}_{\tau\sim\pi_\theta}\left[\log \pi_\theta(\tau \mid x_N, c) - \log \pi_{\mathrm{ref}}(\tau \mid x_N, c) - \beta R(c,\tau) + \log Z_\beta(x_N, c)\right]. \end{aligned} \tag{8}$$

At the optimum, matching $\pi_\theta(\cdot \mid x_N, c)$ to $q^\star_\beta(\cdot \mid x_N, c)$ requires the trajectory-level residual between the current policy score, the reference policy score, the reward energy, and the partition term to be minimized. Motivated by trajectory balance Malkin et al. (2022), we enforce this consistency with a squared trajectory-level residual:

$$\mathcal{L}^{\mathrm{orig}}_{\mathrm{TB}} = \mathbb{E}_{\tau\sim\pi_\theta(\cdot|x_N,c)}\left[\left(\log Z_\beta(x_N, c) + \log \pi_\theta(\tau \mid x_N, c) - \log \pi_{\mathrm{ref}}(\tau \mid x_N, c) - \beta R(c,\tau)\right)^2\right]. \tag{9}$$

This allocates probability across multiple preferred trajectories, giving positive optimization signals to non-maximum but high-quality generations and preserving multiple plausible modes per prompt.

4.2 FROM REWARDFLOW THEORY TO IMAGE FLOW OPTIMIZATION

External Step-Wise KL Regularization. Although the theoretical reward-induced trajectory distribution is defined relative to the pretrained measure π_{ref}, inserting its full log-density into

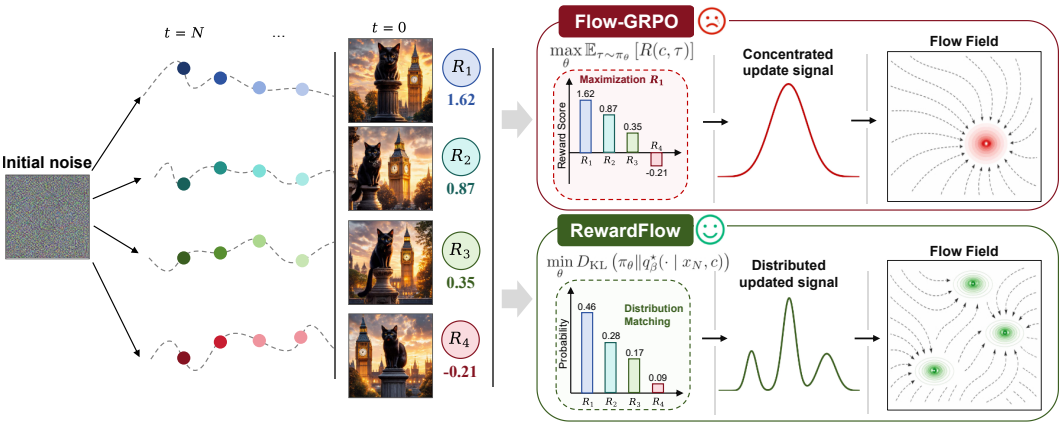

Figure 2: **Overview of RewardFlow.** From initial noise, the flow-matching model samples a group of trajectories, evaluates their final samples, and normalizes rewards within the group. **(a)** Flow-GRPO maximizes scalar rewards, concentrating updates on the highest-reward sample and driving the flow field into a narrow mode. **(b)** RewardFlow matches the reward-induced distribution, assigning probability mass to multiple preferred trajectories, including non-maximum but high-quality ones, thus producing a broader flow field that covers diverse high-quality modes.

the squared trajectory-balance residual is unstable: accumulation across denoising steps and high-dimensional latents can overwhelm rewards, favoring reference-density offset fitting rather than reward-induced probability calibration. RewardFlow therefore separates reward calibration via trajectory balance from pretrained-prior preservation via external step-wise KL regularization.

Flow-based image generation is sensitive to denoising dynamics, where aggressive kernel updates may cause prior drift, artifacts, and reward overfitting. We therefore constrain each transition toward its frozen pretrained counterpart. For Gaussian kernels with fixed variance σ_n^2, the KL reduces to the scaled squared distance between transition means:

$$D_{\mathrm{KL}}\left(K_\theta^n \| K_{\mathrm{ref}}^n\right) = \frac{\|\mu_\theta^n(x_n, c) - \mu_{\mathrm{ref}}^n(x_n, c)\|_2^2}{2\sigma_n^2}. \tag{10}$$

This external KL can be equivalently rewritten as a reference-density $\pi\mathrm{ref}$ term in the trajectory-balance target, as proved in Appendix E.

Learnable Prompt-Conditioned Partition Function. The exact partition function $Z_\beta(x_N, c)$ in equation 6 depends on both the prompt and initial noise. We derive its optimal prompt-only projection from the original trajectory-balance objective. Let $\Delta_\theta(x_N, \tau, c) = \log \pi_\theta(\tau \mid x_N, c) - \log \pi_{\mathrm{ref}}(\tau \mid x_N, c) - \beta R(c, \tau)$. For a fixed policy π_θ, the optimal prompt-level log-normalizer is

$$z_\theta^\dagger(c) = \arg\min_{z\in\mathbb{R}} \mathbb{E}_{\substack{x_N\sim p_N \\ \tau\sim\pi_\theta(\cdot|x_N,c)}} \left[\left(z + \Delta_\theta(x_N, \tau, c)\right)^2\right] = -\mathbb{E}_{\substack{x_N\sim p_N \\ \tau\sim\pi_\theta(\cdot|x_N,c)}} \left[\Delta_\theta(x_N, \tau, c)\right], \tag{11}$$

where $p_N = \mathcal{N}(0, I)$ and $\dagger$ denotes the optimum for fixed θ. At the ideal solution $\pi_{\theta^\star}(\cdot \mid x_N, c) = q_\beta^\star(\cdot \mid x_N, c)$, we have $\Delta_{\theta^\star}(x_N, \tau, c) = -\log Z_\beta(x_N, c)$, yielding $z_{\theta^\star}^\dagger(c) = \mathbb{E}_{x_N\sim p_N}\left[\log Z_\beta(x_N, c)\right]$. We parameterize this projection as $z_\phi(c) = h_\phi(e(c))$, where $e(c)$ is the pooled text embedding and h_ϕ is a lightweight MLP that outputs a scalar log-normalizer. In practice, $z_\phi(c)$ is learned through the trajectory-balance loss, where sampled rewards and trajectory scores provide a rollout-derived supervision signal. Thus, h_ϕ amortizes the prompt-level normalizer across training prompts rather than predicting it without supervision. Since the same $z_\phi(c)$ is shared across all trajectories of a prompt, it can absorb only their common offset, while the flow policy must still match trajectory-dependent reward differences.

Warm Start for the Partition Head. RewardFlow decouples reward calibration from pretrained-prior preservation and uses prompt-wise normalized rewards $\hat{r}_i = (r_i - \mu_c)/(\sigma_c + \epsilon)$, where μ_c and σ_c are the within-prompt reward mean and standard deviation. The trajectory-balance residual is $\delta_i = z_\phi(c) + s_\theta(\tau_i, c) - \beta\hat{r}_i$, where $s_\theta(\tau_i, c)$ is the average trajectory log-probability under the current

denoising policy π_θ. Randomly initializing $z_\phi(c)$ can introduce a large residual offset from the scale mismatch between trajectory scores and normalized rewards, causing early optimization to focus on offset cancellation rather than reward-induced probability allocation. We therefore warm-start $z_\phi(c)$ under the old policy before updating the flow policy, such that

$$\mathbb{E}\left[z_\phi(c) + s_{\text{old}}(\tau, c) - \beta\hat{r}\right] \approx 0. \tag{12}$$

Residual centering stabilizes early training without changing the reward-induced target distribution.

4.3 Final Objective

RewardFlow Training Objective. Given a prompt c and G sampled trajectories $\{\tau_i\}_{i=1}^G$, RewardFlow jointly optimizes the flow policy π_θ and the prompt-conditioned partition head z_ϕ with:

$$\begin{aligned}\mathcal{L}_{\text{RewardFlow}} = &\underbrace{\frac{1}{G}\sum_{i=1}^{G} w_i\left[z_\phi(c) + s_\theta(\tau_i, c) - \beta\hat{r}_i\right]^2}_{\mathcal{L}_{\text{TB}}} \\ &+ \lambda_{\text{KL}}\underbrace{\frac{1}{2GN}\sum_{i=1}^{G}\sum_{n=1}^{N}\sigma_n^{-2}\left\|\mu_\theta^n(x_n^i, c) - \mu_{\text{ref}}^n(x_n^i, c)\right\|_2^2}_{\mathcal{L}_{\text{KL}}}.\end{aligned} \tag{13}$$

Here, the trajectory score is defined as $s_\theta(\tau_i, c) = \frac{1}{N}\sum_{n=1}^{N}\log K_\theta^n(x_{n-1}^i \mid x_n^i, c)$, and the detached clipped importance weight is $w_i = \text{sg}\left[\text{clip}\left(\exp\left(s_\theta(\tau_i, c) - s_{\text{old}}(\tau_i, c)\right), 1-\epsilon_{\text{clip}}, 1+\epsilon_{\text{clip}}\right)\right]$ and $\text{sg}[\cdot]$ denotes the stop-gradient operator.

5 Experiments

5.1 Experimental Setup

Foundation Models and Datasets. We evaluate RewardFlow on two flow-based text-to-image models: Stable Diffusion 3.5 Medium (SD3.5-M) Esser et al. (2024) and FLUX.1-dev Labs (2024). For human preference alignment, we use the official HPDv2 prompts Wu et al. (2023), which contain 107K training prompts and 3,200 evaluation prompts evenly divided into Animation, Concept-art, Painting, and Photo. For compositional alignment and text rendering, GenEval Ghosh et al. (2023) and OCR Du et al. (2009) are trained and evaluated on their task-specific datasets. All methods use the same training and evaluation splits for each task to ensure fair comparison.

Table 1: Comparison results under preference **multi-reward training** with three human-preference objectives. All methods are trained with **HPS-v2.1**, **Pick Score**, and **ImageReward** with a ratio of 1:1:1, and evaluated on six metrics. Relative improvements are computed against the corresponding base model in each block.

Method	HPS-v2.1↑	Pick Score↑	ImageReward↑	GenEval↑	OCR↑	Aesthetic↑
SD3.5-M Backbone						
SD3.5-M Esser et al. (2024)	0.293 (+0.0%)	0.224 (+0.0%)	1.065 (+0.0%)	0.637 (+0.0%)	0.569 (+0.0%)	5.967 (+0.0%)
FlowGRPO Liu et al. (2025b)	0.326 (+11.3%)	0.229 (+2.2%)	1.526 (+43.3%)	0.714 (+12.1%)	0.588 (+3.3%)	6.226 (+4.3%)
MixGRPO Li et al. (2025a)	0.338 (+15.4%)	0.233 (+4.0%)	1.591 (+49.4%)	0.703 (+10.4%)	0.608 (+6.9%)	6.285 (+5.3%)
RewardFlow (**Ours**)	**0.362** (+23.5%)	**0.238** (+6.3%)	**1.650** (+54.9%)	**0.734** (+15.2%)	**0.612** (+7.6%)	**6.470** (+8.4%)
FLUX.1-dev Backbone						
FLUX.1-dev Labs (2024)	0.302 (+0.0%)	0.228 (+0.0%)	1.061 (+0.0%)	0.635 (+0.0%)	0.590 (+0.0%)	6.226 (+0.0%)
FlowGRPO Liu et al. (2025b)	0.344 (+13.9%)	0.229 (+0.4%)	1.595 (+50.3%)	0.726 (+14.3%)	0.743 (+25.9%)	6.405 (+2.9%)
MixGRPO Li et al. (2025a)	0.366 (+21.2%)	**0.235** (+3.1%)	1.604 (+51.2%)	**0.730** (+15.0%)	**0.746** (+26.4%)	6.623 (+6.4%)
RewardFlow (**Ours**)	**0.367** (+21.5%)	**0.235** (+3.1%)	**1.661** (+56.6%)	**0.764** (+20.3%)	**0.756** (+28.1%)	**6.656** (+6.9%)

Reward Models and Evaluation Metrics. We consider six evaluation metrics spanning three complementary aspects: HPS-v2.1 Wu et al. (2023), PickScore Kirstain et al. (2023), ImageReward Xu et al. (2023), and Aesthetic Score Schuhmann for human-preference alignment, GenEval for compositional alignment and OCR for text-rendering accuracy. We study both single-reward

Table 2: Comparison results under mixed **multi-reward training** across preference, compositional, and text-rendering objectives. All methods are trained with **HPS-v2.1**, **GenEval**, and **OCR** with a ratio of 1:1:1, and evaluated on six metrics. Relative improvements are computed against the corresponding base model in each block.

Method	HPS-v2.1↑	GenEval↑	OCR↑	Pick Score↑	ImageReward↑	Aesthetic↑
SD3.5-M Backbone						
SD3.5-M Esser et al. (2024)	0.293 (+0.0%)	0.637 (+0.0%)	0.569 (+0.0%)	0.224 (+0.0%)	1.065 (+0.0%)	5.967 (+0.0%)
FlowGRPO Liu et al. (2025b)	0.294 (+0.3%)	0.816 (+28.1%)	0.860 (+51.1%)	0.223 (-0.4%)	1.075 (+0.9%)	5.990 (+0.4%)
RewardFlow (**Ours**)	**0.355** (+21.2%)	**0.832** (+30.6%)	**0.902** (+58.5%)	**0.231** (+3.1%)	**1.347** (+26.5%)	**6.304** (+5.6%)
FLUX.1-dev Backbone						
FLUX.1-dev Labs (2024)	0.302 (+0.0%)	0.635 (+0.0%)	0.590 (+0.0%)	0.228 (+0.0%)	1.061 (+0.0%)	6.226 (+0.0%)
FlowGRPO Liu et al. (2025b)	0.319 (+5.6%)	0.903 (+42.2%)	0.926 (+56.9%)	0.229 (+0.4%)	1.342 (+26.5%)	6.365 (+2.2%)
RewardFlow (**Ours**)	**0.363** (+20.2%)	**0.922** (+45.2%)	**0.945** (+60.2%)	**0.231** (+1.3%)	**1.527** (+43.9%)	**6.527** (+4.8%)

and multi-reward training. For single-reward training, we optimize HPS-v2.1, GenEval, or OCR individually. For multi-reward training, we consider two settings: (i) preference-oriented training with HPS-v2.1, PickScore, and ImageReward, and (ii) heterogeneous training with HPS-v2.1, GenEval, and OCR, both using an equal 1:1:1 ratio.

Training and Comparison Configuration. All experiments are conducted on 8 NVIDIA A100 GPUs. We use a prompt batch size of 9 for SD3.5-M and 3 for FLUX.1-dev, with a group size of 24 for both models. Each model is trained for 600 epochs under both single-reward and multi-reward settings. We set the reward coefficient $\beta = 1$ and the step-wise KL regularization weight $\lambda_{\mathrm{KL}} = 0.1$. We compare RewardFlow with Flow-GRPO under identical configurations and report the pretrained base models as references. Implementation details are provided in Appendix D.

5.2 Main Results

Quantitative analysis across reward settings. Tables 1, 2, and 3 evaluate preference multi-reward, mixed multi-reward, and single-reward training. When optimizing HPS-v2.1, GenEval, or OCR individually, RewardFlow consistently outperforms Flow-GRPO on both backbones in the training metric and cross-metric transfer to preference and image-quality metrics. With multiple correlated preference rewards, RewardFlow consistently outperforms Flow-GRPO and MixGRPO, raising HPS-v2.1 from 0.326 to 0.362 and ImageReward from 1.526 to 1.650 over Flow-GRPO on SD3.5-M. On FLUX.1-dev, it leads in HPS-v2.1, ImageReward, and Aesthetic while matching the best Pick Score. These gains persist when combining heterogeneous HPS-v2.1, GenEval, and OCR rewards without overspecializing to one objective.

Qualitative analysis of generated images. We further compare visual results in Figure 3 and Appendix B. The qualitative comparison is consistent with the quantitative results. Flow-GRPO often tends to emphasize reward-favored visual patterns, leading to more similar compositions, saturated colors, and over-enhanced textures. As a result, samples generated under the same prompt become visually homogenized, with reduced diversity in layout, style, and appearance. In contrast, RewardFlow produces more balanced images with clearer object structures, more natural appearance, and better visual coherence. It also preserves richer visual diversity across samples, indicating that reward-distribution matching can improve preference alignment without collapsing toward a narrow set of high-reward modes. This difference reflects the advantage of reward-distribution matching.

5.3 Analysis

Analysis on diversity, generalization, and multi-reward optimization. We revisit the three limitations discussed in the introduction. Figure 1 shows that RewardFlow preserves richer same-prompt variations in layout, style, and appearance than Flow-GRPO. DreamSim Fu et al. (2023), FID Heusel et al. (2017), BFID Dombrowski et al. (2025), and SSIM Wang et al. (2004) in Table 4 confirm greater perceptual variation and better preservation of the overall generation distribution, indicating reduced mode collapse. For generalization, under HPS-v2.1, GenEval, or OCR single-reward training, RewardFlow consistently outperforms Flow-GRPO on both the optimized metric and held-out preference and image-quality metrics, demonstrating stronger cross-metric transfer. It



Figure 3: **Qualitative results of RewardFlow compared with Flow-GRPO.** Our method maintains diverse generations across different visual layouts and styles, while improving visual quality.

Table 3: Comparison results under the **single-reward training setting**. Within each backbone, the three FlowGRPO/RewardFlow pairs are trained with **HPS-v2.1**, **GenEval**, and **OCR**, respectively, from top to bottom. Relative improvements in parentheses are computed against the corresponding base model. The better result is highlighted in bold.

Method	Task Metric			Preference Score		Image Quality
	HPS-v2.1↑	GenEval↑	OCR Acc.↑	Pick Score↑	ImageReward↑	Aesthetic↑
SD3.5-M Backbone						
SD3.5-M	0.293 (+0.0%)	0.637 (+0.0%)	0.569 (+0.0%)	0.224 (+0.0%)	1.065 (+0.0%)	5.967 (+0.0%)
FlowGRPO	0.360 (+22.9%)	–	–	0.231 (+3.1%)	1.450 (+36.2%)	6.330 (+6.1%)
RewardFlow (Ours)	**0.368** (+25.6%)	–	–	**0.232** (+3.6%)	**1.509** (+41.7%)	**6.468** (+8.4%)
FlowGRPO	–	0.837 (+31.4%)	–	0.224 (+0.0%)	1.096 (+2.9%)	5.946 (-0.4%)
RewardFlow (Ours)	–	**0.845** (+32.7%)	–	**0.226** (+0.9%)	**1.110** (+4.2%)	**5.993** (+0.4%)
FlowGRPO	–	–	0.926 (+62.7%)	0.223 (-0.4%)	1.059 (-0.6%)	5.887 (-1.3%)
RewardFlow (Ours)	–	–	**0.935** (+64.3%)	**0.225** (+0.4%)	**1.116** (+4.8%)	**6.037** (+1.2%)
FLUX.1-dev Backbone						
FLUX.1-dev	0.302 (+0.0%)	0.635 (+0.0%)	0.590 (+0.0%)	0.228 (+0.0%)	1.061 (+0.0%)	6.226 (+0.0%)
FlowGRPO	0.370 (+22.5%)	–	–	0.229 (+0.4%)	1.457 (+37.3%)	6.408 (+2.9%)
RewardFlow (Ours)	**0.377** (+24.8%)	–	–	**0.231** (+1.3%)	**1.516** (+42.9%)	**6.505** (+4.5%)
FlowGRPO	–	0.956 (+50.6%)	–	0.223 (-2.2%)	1.080 (+1.8%)	6.138 (-1.4%)
RewardFlow (Ours)	–	**0.960** (+51.2%)	–	**0.229** (+0.4%)	**1.132** (+6.7%)	**6.198** (-0.4%)
FlowGRPO	–	–	0.933 (+58.1%)	0.227 (-0.4%)	1.124 (+5.9%)	6.177 (-0.8%)
RewardFlow (Ours)	–	–	**0.948** (+60.7%)	**0.230** (+0.9%)	**1.165** (+9.8%)	**6.338** (+1.8%)

also remains consistently strong under preference-based and heterogeneous HPS-v2.1/GenEval/OCR multi-reward training, whereas Flow-GRPO degrades more when combining reward objectives. Overall, reward-distribution matching improves alignment while preserving diversity, cross-metric transfer, and multi-reward stability. Additional qualitative results appear in Appendix B.

User Study. To assess whether quantitative gains translate into human-perceived improvements, participants compare images from SD3.5-M, Flow-GRPO, and RewardFlow on Prompt Alignment, Visual Quality, Generation Diversity, and Overall Preference. Table 5 shows that RewardFlow achieves the highest preference ratio across all four aspects, with particularly strong advantages in Diversity and Overall Preference, confirming its strong visual quality and text-image alignment.

Computational efficiency. RewardFlow introduces no extra overhead over Flow-GRPO, using the same sampling process, reward-model evaluation, and training budget without additional denoising

Table 4: Comparison of diversity metrics under different methods.

Method	Diversity			
	DreamSim↑	FID↓	BFID↑	SSIM↑
Flow-GRPO	0.163	34.040	0.097	0.198
DiverseGRPO Liu et al. (2025a)	0.185	29.820	0.165	0.210
RewardFlow **(Ours)**	**0.196**	**22.718**	**0.385**	**0.356**

Table 5: **User study results.** Values indicate the fraction of times each method is selected as the best.

Method	Prompt Align. ↑	Visual Quality ↑	Diversity ↑	Overall Pref. ↑
SD3.5-M	0.076	0.054	0.072	0.040
Flow-GRPO	0.156	0.122	0.116	0.092
RewardFlow **(Ours)**	**0.768**	**0.824**	**0.812**	**0.868**

steps or reward calls. It therefore improves preference alignment while maintaining the same computational cost and comparable training speed.

5.4 Ablation Studies

Ablation of reward sharpness. Table 6 studies the reward sharpness coefficient β. The best performance is achieved at $\beta = 1.0$. Smaller values oversmooth the reward-induced distribution and weaken the alignment signal, while larger values make it overly sharp and concentrate probability on a few high-reward trajectories. We therefore use $\beta = 1.0$ by default.

Ablation of step-wise KL regularization. Table 7 studies the effect of the step-wise KL weight λ_{KL}. A moderate KL weight achieves the best performance, with $\lambda_{\mathrm{KL}} = 0.10$. A weaker constraint allows excessive policy drift, whereas stronger constraints overly restrict policy updates and limit reward gains. We therefore adopt $\lambda_{\mathrm{KL}} = 0.10$ as the default. To further examine the role of this KL anchor and alternative prior-preservation designs, we report additional ablations in Table 8. Removing the step-wise KL causes a substantial degradation, as RewardFlow optimizes a trajectory-balance residual rather than directly maximizing reward; without the KL anchor, the policy can drift from the pretrained flow prior without necessarily improving reward. In contrast, directly incorporating π_{ref} into the trajectory-balance residual is also inferior, since its accumulated log-density has large scale and high trajectory-wise variance, which can dominate residual fitting.

Table 6: Ablation study on the reward sharpness coefficient β.

β	HPS-v2.1↑	Pick Score↑	ImageReward↑	Aesthetic↑
0.500	0.353	0.231	1.421	6.240
1.000	**0.368**	**0.232**	**1.509**	**6.468**
1.500	0.335	0.226	1.416	6.079
2.000	0.317	0.215	1.401	5.700

Table 7: Ablation study on the step-wise KL regularization weight λ_{KL}.

λ_{KL}	HPS-v2.1↑	Pick Score↑	ImageReward↑	Aesthetic↑
0.050	0.363	0.229	1.470	6.326
0.100	**0.368**	**0.232**	**1.509**	**6.468**
0.150	0.347	0.231	1.432	6.282
0.200	0.344	0.228	1.420	6.218

Ablation of partition function and initialization. Table 8 evaluates the partition function $z_\phi(c)$ and its initialization under the same step-wise KL. The prompt-conditioned $z_\phi(c)$ performs best, as it absorbs prompt-dependent offsets in the trajectory-balance residual and focuses optimization on relative reward-induced probability allocation rather than cross-prompt likelihood-scale differences. Removing $z_\phi(c)$ or replacing it with a constant degrades performance. Moreover, warm initialization improves results over random initialization by providing a stable initial residual for early training.

Table 8: Ablation results on RewardFlow optimization designs.

Ablation	Partition Function	Initialization	Reference Prior	HPS-v2.1↑	Pick Score↑	ImageReward↑	Aesthetic↑
RewardFlow (Ours)	Prompt-Conditioned $z_\phi(c)$	Warm Start	Step-Wise KL	**0.368**	**0.232**	**1.509**	**6.468**
Partition Function	Prompt-Conditioned $z_\phi(c)$	Random Init.	Step-Wise KL	0.363	0.231	1.465	6.385
	No Partition $z(c) = 0$	–	Step-Wise KL	0.334	0.226	1.325	6.157
	Reward-Based Constant $z(c) = z_0$	Fixed	Step-Wise KL	0.352	0.229	1.405	6.271
Reference Prior	Prompt-Conditioned $z_\phi(c)$	Warm Start	None	0.270	0.210	1.173	5.695
	Prompt-Conditioned $z_\phi(c)$	Warm Start	Internal π_{ref} in TB	0.328	0.217	1.276	5.982

6 Conclusion

In this work, we propose RewardFlow, a reward-distribution matching framework for post-training flow-based image generation. Instead of optimizing a single scalar reward, RewardFlow constructs a prompt-wise reward-induced distribution from multiple generated samples and aligns the flow policy with this distribution, thereby preserving multiple preferred modes rather than over-concentrating on a few reward-favored samples. With a warm-started learnable prompt-conditioned function and a step-wise KL constraint, RewardFlow stabilizes distribution matching while maintaining the pretrained flow prior. Experiments under single-reward and multi-reward settings demonstrate that RewardFlow achieves stronger preference alignment, better generalization, and more diverse generations than Flow-GRPO, showing the promise of distribution-aware post-training for generative models.

AI USE STATEMENT

In this work, we used generative AI tools to improve the grammar, clarity, and readability of the manuscript. We did not use generative AI tools to develop the RewardFlow framework, propose or refine research hypotheses, formulate mathematical claims, develop or write proofs, design research methodology or experiments, implement methods, conduct qualitative analyses, or interpret experimental results.

We have reviewed all AI-assisted text. In particular, the authors checked the revised wording against the technical descriptions, mathematical formulations, and experimental results to ensure that it accurately reflects the work and preserves the intended scientific meaning. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

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

Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J Hu, Mo Tiwari, and Emmanuel Bengio. Gflownet foundations. *Journal of Machine Learning Research*, 24(210):1–55, 2023.

Kaijie Chen, Zhiyang Xu, Ying Shen, Zihao Lin, Yuguang Yao, and Lifu Huang. Superflow: Training flow matching models with rl on the fly. *arXiv preprint arXiv:2512.17951*, 2025.

Haoyou Deng, Keyu Yan, Chaojie Mao, Xiang Wang, Yu Liu, Changxin Gao, and Nong Sang. Densegrpo: From sparse to dense reward for flow matching model alignment. *arXiv preprint arXiv:2601.20218*, 2026.

Zheng Ding and Weirui Ye. Treegrpo: Tree-advantage grpo for online rl post-training of diffusion models. *arXiv preprint arXiv:2512.08153*, 2025.

Mischa Dombrowski, Weitong Zhang, Sarah Cechnicka, Hadrien Reynaud, and Bernhard Kainz. Image generation diversity issues and how to tame them. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 3029–3039, 2025.

Yuning Du, Chenxia Li, Ruoyu Guo, Xiaoting Yin, Weiwei Liu, Jun Zhou, Yifan Bai, Zilin Yu, Yehua Yang, Qingqing Dang, et al. Pp-ocr: a practical ultra lightweight ocr system; 2020. *Preprint*, 2009.

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.

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

Peng Gao, Le Zhuo, Dongyang Liu, Ruoyi Du, Xu Luo, Longtian Qiu, Yuhang Zhang, Chen Lin, Rongjie Huang, Shijie Geng, et al. Lumina-t2x: Transforming text into any modality, resolution, and duration via flow-based large diffusion transformers. *arXiv preprint arXiv:2405.05945*, 2024.

Dhruba Ghosh, Hanna Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment, 2023. URL https://arxiv.org/abs/2310.11513.

Xiaoxuan He, Siming Fu, Yuke Zhao, Wanli Li, Jian Yang, Dacheng Yin, Fengyun Rao, and Bo Zhang. Tempflow-grpo: When timing matters for grpo in flow models. *arXiv preprint arXiv:2508.04324*, 2025.

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

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.

Black Forest Labs. Flux. `https://github.com/black-forest-labs/flux`, 2024.

Jeongjae Lee and Jong Chul Ye. Pcpo: Proportionate credit policy optimization for aligning image generation models. *arXiv preprint arXiv:2509.25774*, 2025.

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*, 2025a.

Yuming Li, Yikai Wang, Yuying Zhu, Zhongyu Zhao, Ming Lu, Qi She, and Shanghang Zhang. Branchgrpo: Stable and efficient grpo with structured branching in diffusion models. *arXiv preprint arXiv:2509.06040*, 2025b.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Henglin Liu, Huijuan Huang, Jing Wang, Chang Liu, Xiu Li, and Xiangyang Ji. Diversegrpo: Mitigating mode collapse in image generation via diversity-aware grpo. *arXiv preprint arXiv:2512.21514*, 2025a.

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*, 2025b.

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

Zexiang Liu, Xianglong He, and Yangguang Li. Advances in grpo for generation models: A survey. *arXiv preprint arXiv:2603.06623*, 2026.

Yifu Luo, Penghui Du, Bo Li, Sinan Du, Tiantian Zhang, Yongzhe Chang, Kai Wu, Kun Gai, and Xueqian Wang. Sample by step, optimize by chunk: Chunk-level grpo for text-to-image generation. *arXiv preprint arXiv:2510.21583*, 2025.

Qiang Lyu, Zicong Chen, Chongxiao Wang, Haolin Shi, Shibo Gao, Ran Piao, Youwei Zeng, Jianlou Si, Fei Ding, Jing Li, et al. Multi-grpo: Multi-group advantage estimation for text-to-image generation with tree-based trajectories and multiple rewards. *arXiv preprint arXiv:2512.00743*, 2025.

Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in gflownets. *Advances in Neural Information Processing Systems*, 35: 5955–5967, 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.

Chrisoph Schuhmann. Laion aesthetics, aug 2022. *URL https://laion.ai/blog/laion-aesthetics*.

Yawen Shao, Jie Xiao, Kai Zhu, Yu Liu, Wei Zhai, Yang Cao, and Zheng-Jun Zha. Value-anchored group policy optimization for flow models.

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

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Xiaoshi Wu, Keqiang Sun, Feng Zhu, Rui Zhao, and Hongsheng Li. Better aligning text-to-image models with human preference. *arXiv preprint arXiv:2303.14420*, 1(3), 2023.

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

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.

Ruizhe Zhong, Jiesong Lian, Xiaoyue Mi, Zixiang Zhou, Yuan Zhou, Qinglin Lu, and Junchi Yan. Euphonium: Steering video flow matching via process reward gradient guided stochastic dynamics. *arXiv preprint arXiv:2602.04928*, 2026.

Yujie Zhou, Pengyang Ling, Jiazi Bu, Yibin Wang, Yuhang Zang, Jiaqi Wang, Li Niu, and Guangtao Zhai. Fine-grained grpo for precise preference alignment in flow models. *arXiv preprint arXiv:2510.01982*, 2025.

Xuekai Zhu, Daixuan Cheng, Dinghuai Zhang, Hengli Li, Kaiyan Zhang, Che Jiang, Youbang Sun, Ermo Hua, Yuxin Zuo, Xingtai Lv, et al. Flowrl: Matching reward distributions for llm reasoning. *arXiv preprint arXiv:2509.15207*, 2025.

A ALGORITHMIC DETAILS OF REWARDFLOW

Algorithm 1 RewardFlow Training Process

Require: Frozen policy π_{ref}, trainable policy π_θ, reward model r, decoder $\mathcal{D}$, partition head $z_\phi(c)$, group size G, steps N, β, λ_{KL}, ϵ_{clip}

Ensure: Optimized π_θ and z_ϕ

Initialize $\pi_\theta \leftarrow \pi_{\text{ref}}$, freeze π_{ref}, and warm-start z_ϕ so that $\mathbb{E}[z_\phi(c) + s_{\text{old}}(\tau, c) - \beta\hat{r}] \approx 0$
Define $S_\psi(\tau_i, c) = \frac{1}{N}\sum_{n=1}^{N} \log K_\psi^n(x_{n-1}^i \mid x_n^i, c)$
for each training iteration **do**
  Sample prompt c and set $\pi_{\text{old}} \leftarrow \pi_\theta$
  Sample G trajectories $\tau_i = (x_N^i, \ldots, x_0^i) \sim \pi_{\text{old}}(\cdot \mid x_N^i, c)$, decode $y_i = \mathcal{D}(x_0^i)$, compute $r_i = r(y_i, c)$ and $s_{\text{old}}^i = S_{\text{old}}(\tau_i, c)$
  Normalize rewards: $\hat{r}_i = (r_i - \mu_c)/(\sigma_c + \epsilon)$, with $\mu_c = \frac{1}{G}\sum_{i=1}^{G} r_i$
  for each optimization epoch **do**
    Compute $s_\theta^i = S_\theta(\tau_i, c)$ and $w_i = \text{sg}\left[\text{clip}\left(e^{s_\theta^i - s_{\text{old}}^i}, 1 - \epsilon_{\text{clip}}, 1 + \epsilon_{\text{clip}}\right)\right]$
    Minimize

$$\mathcal{L}_{\text{RewardFlow}} = \frac{1}{G}\sum_{i=1}^{G} w_i \left(z_\phi(c) + s_\theta^i - \beta\hat{r}_i\right)^2 + \lambda_{\text{KL}} \frac{1}{GN}\sum_{i=1}^{G}\sum_{n=1}^{N} \frac{\left\|\mu_\theta^n(x_n^i, c) - \mu_{\text{ref}}^n(x_n^i, c)\right\|_2^2}{2\sigma_n^2}$$

    with respect to θ, ϕ
  end for
end for

B ADDITIONAL VISUAL RESULTS OF REWARDFLOW



Figure 4: Comparison of the visualization results of Flow-GRPO and RewardFlow.



Figure 5: Additional qualitative results of RewardFlow. Our method produces visually appealing and diverse images across various categories and styles, demonstrating strong generation quality, detailed structures, and broad prompt generalization.



Figure 6: Additional qualitative results of RewardFlow. Our method produces visually appealing and diverse images across various categories and styles, demonstrating strong generation quality, detailed structures, and broad prompt generalization.



Figure 7: Additional qualitative results of RewardFlow. Our method produces visually appealing and diverse images across various categories and styles, demonstrating strong generation quality, detailed structures, and broad prompt generalization.

C LIMITATION

RewardFlow still has several limitations. First, it constructs the reward-induced distribution from a finite group of sampled trajectories, so some valid high-quality modes may be missed when the group size is limited. Additionally, the method relies on external reward models, and their biases or failure cases may be inherited by the aligned flow policy. Overall, these limitations suggest that improving trajectory coverage and developing more robust reward modeling are important directions for making RewardFlow more reliable and generalizable.

D IMPLEMENTATION DETAILS

We keep the training and sampling configurations consistent across Flow-GRPO and RewardFlow. Both SD3.5-M and FLUX.1-dev use the AdamW optimizer, a learning rate of 1×10^{-5}, a weight decay of 1×10^{-4}, and a training resolution of 512×512. The two backbones only differ in their model-specific sampling settings. For SD3.5-M, we use $N = 10$ denoising steps with CFG scale 4.5 during training, and $N = 40$ denoising steps with CFG scale 4.5 during evaluation. For FLUX.1-dev, we use $N = 6$ denoising steps with guidance scale 3.5 during training, and $N = 28$ denoising steps with guidance scale 4.5 during evaluation. Apart from these backbone-specific settings, all methods share the same prompt source, reward-model evaluation pipeline, training budget, and evaluation protocol.

E CONNECTION BETWEEN EXTERNAL KL AND REFERENCE-AWARE TRAJECTORY BALANCE

In this appendix, we show that the external step-wise KL regularization can be interpreted as a coefficient-controlled reference-density term in the trajectory-balance objective. This explains why RewardFlow can preserve the pretrained flow prior without directly inserting the full reference log-density into the squared trajectory-balance residual.

For clarity, we omit the initial noise condition x_N in the notation. Given a denoising trajectory $\tau = (x_N, \dots, x_0)$, we define the averaged trajectory log-probability under policy π_ψ as

$$s_\psi(\tau, c) = \frac{1}{N} \sum_{n=1}^{N} \log K_\psi^n(x_{n-1} \mid x_n, c), \qquad \psi \in \{\theta, \mathrm{ref}\}. \tag{14}$$

Since both the current transition kernel and the reference transition kernel are Gaussian with the same variance,

$$K_\theta^n(\cdot \mid x_n, c) = \mathcal{N}\big(\mu_\theta^n(x_n, c), \sigma_n^2 I\big), \qquad K_{\mathrm{ref}}^n(\cdot \mid x_n, c) = \mathcal{N}\big(\mu_{\mathrm{ref}}^n(x_n, c), \sigma_n^2 I\big), \tag{15}$$

the step-wise KL has the closed form

$$\begin{aligned} D_{\mathrm{KL}}\left(K_\theta^n \| K_{\mathrm{ref}}^n\right) &= \mathbb{E}_{x_{n-1} \sim K_\theta^n}\left[\log K_\theta^n(x_{n-1} \mid x_n, c) - \log K_{\mathrm{ref}}^n(x_{n-1} \mid x_n, c)\right] \\ &= \frac{\|\mu_\theta^n(x_n, c) - \mu_{\mathrm{ref}}^n(x_n, c)\|_2^2}{2\sigma_n^2}. \end{aligned} \tag{16}$$

Therefore, after averaging over denoising steps and trajectories, the external KL regularizer satisfies

$$\begin{aligned} \mathcal{L}_{\mathrm{KL}} &= \lambda_{\mathrm{KL}} \mathbb{E}_{\tau \sim \pi_\theta}\left[\frac{1}{N} \sum_{n=1}^{N} D_{\mathrm{KL}}\left(K_\theta^n \| K_{\mathrm{ref}}^n\right)\right] \\ &= \lambda_{\mathrm{KL}} \mathbb{E}_{\tau \sim \pi_\theta}\left[s_\theta(\tau, c) - s_{\mathrm{ref}}(\tau, c)\right]. \end{aligned} \tag{17}$$

Now consider the practical reward-matching objective without the internal reference-density term:

$$\mathcal{J}_0 = D_{\mathrm{KL}}\left(\pi_\theta(\tau \mid c) \| q_\beta(\tau \mid c)\right), \qquad q_\beta(\tau \mid c) \propto \exp(\beta \hat{r}(\tau, c)). \tag{18}$$

Expanding this KL gives

$$\mathcal{J}_0 = \mathbb{E}_{\tau \sim \pi_\theta}\left[s_\theta(\tau, c) - \beta \hat{r}(\tau, c)\right] + \log Z_\beta(c), \tag{19}$$

where $Z_\beta(c)$ is the corresponding normalizing constant. Adding the external KL term yields

$$\begin{aligned}\mathcal{J}_\lambda &= \mathcal{J}_0 + \lambda_{\mathrm{KL}}\mathbb{E}_{\tau\sim\pi_\theta}\left[s_\theta(\tau,c) - s_{\mathrm{ref}}(\tau,c)\right] \\ &= \mathbb{E}_{\tau\sim\pi_\theta}\left[(1+\lambda_{\mathrm{KL}})s_\theta(\tau,c) - \lambda_{\mathrm{KL}}s_{\mathrm{ref}}(\tau,c) - \beta\hat{r}(\tau,c)\right] + \log Z_\beta(c) \\ &= (1+\lambda_{\mathrm{KL}})\mathbb{E}_{\tau\sim\pi_\theta}\left[s_\theta(\tau,c) - \frac{\lambda_{\mathrm{KL}}}{1+\lambda_{\mathrm{KL}}}s_{\mathrm{ref}}(\tau,c) - \frac{\beta}{1+\lambda_{\mathrm{KL}}}\hat{r}(\tau,c)\right] + \log Z_\beta(c). \end{aligned} \tag{20}$$

Since $1+\lambda_{\mathrm{KL}} > 0$, the above objective is equivalent, up to a positive scaling factor and a normalizing constant, to matching the following reference-aware reward-induced distribution:

$$q^\star_{\beta,\lambda}(\tau \mid c) = \frac{\exp(\alpha_{\mathrm{ref}}s_{\mathrm{ref}}(\tau,c) + \beta_\lambda\hat{r}(\tau,c))}{Z_{\beta,\lambda}(c)}, \tag{21}$$

where

$$\alpha_{\mathrm{ref}} = \frac{\lambda_{\mathrm{KL}}}{1+\lambda_{\mathrm{KL}}}, \qquad \beta_\lambda = \frac{\beta}{1+\lambda_{\mathrm{KL}}}. \tag{22}$$

Equivalently, since $s_{\mathrm{ref}}(\tau,c) = \log\pi_{\mathrm{ref}}(\tau \mid c)$ up to the averaging convention, we can write

$$q^\star_{\beta,\lambda}(\tau \mid c) \propto \pi_{\mathrm{ref}}(\tau \mid c)^{\alpha_{\mathrm{ref}}}\exp(\beta_\lambda\hat{r}(\tau,c))\,. \tag{23}$$

This derivation shows that adding an external KL regularizer is equivalent, at the KL-expanded objective level, to introducing the pretrained reference log-density into the trajectory-balance target with a controllable coefficient α_{ref}. The corresponding reference-aware trajectory-balance residual can be written as

$$\delta_i^{\mathrm{ref}} = z_{\phi,\lambda}(c) + s_\theta(\tau_i,c) - \alpha_{\mathrm{ref}}s_{\mathrm{ref}}(\tau_i,c) - \beta_\lambda\hat{r}_i. \tag{24}$$

In practice, RewardFlow keeps the reference constraint as an external step-wise KL term rather than directly placing s_{ref} inside the squared residual. This avoids the accumulated high-dimensional reference log-density from dominating the trajectory-balance residual, while still preserving the pretrained flow prior through a coefficient-controlled reference constraint.