Flow-GRPO: Training Flow Matching Models via Online RL

Jie Liu[1,3,5*] **Gongye Liu**[2,3*] **Jiajun Liang**[3] **Yangguang Li**[1]
Jiaheng Liu[4] **Xintao Wang**[3] **Pengfei Wan**[3] **Di Zhang**[3] **Wanli Ouyang**[1,5†]
[1]MMLab, CUHK [2]Tsinghua University [3]Kling Team, Kuaishou Technology
[4]Nanjing University [5]Shanghai AI Laboratory
jieliu@link.cuhk.edu.hk wlouyang@ie.cuhk.edu.hk
Code: https://github.com/yifan123/flow_grpo

Abstract

We propose Flow-GRPO, the first method to integrate online policy gradient reinforcement learning (RL) into flow matching models. Our approach uses two key strategies: (1) an ODE-to-SDE conversion that transforms a deterministic Ordinary Differential Equation (ODE) into an equivalent Stochastic Differential Equation (SDE) that matches the original model's marginal distribution at all timesteps, enabling statistical sampling for RL exploration; and (2) a Denoising Reduction strategy that reduces training denoising steps while retaining the original number of inference steps, significantly improving sampling efficiency without sacrificing performance. Empirically, Flow-GRPO is effective across multiple text-to-image tasks. For compositional generation, RL-tuned SD3.5-M generates nearly perfect object counts, spatial relations, and fine-grained attributes, increasing GenEval accuracy from 63% to 95%. In visual text rendering, accuracy improves from 59% to 92%, greatly enhancing text generation. Flow-GRPO also achieves substantial gains in human preference alignment. Notably, very little reward hacking occurred, meaning rewards did not increase at the cost of appreciable image quality or diversity degradation.

1 Introduction

Flow matching [2, 3] models have become dominant in image generation [4, 5] due to their solid theoretical foundations and strong performance in producing high quality images. However, they often struggle with composing complex scenes involving multiple objects, attributes, and relationships [6, 7], as well as text rendering [8]. At the same time, online reinforcement learning (RL) [9] has proven highly effective in enhancing the reasoning capabilities of large language models (LLMs) [10, 11]. While previous research has mainly focused on applying RL to early diffusion-based generative models [12] and offline RL techniques like direct preference optimization [13] for flow-based generative models [14, 15], the potential of online RL in advancing flow matching generative models remains largely unexplored. In this study, we explore how online RL can be leveraged to effectively improve flow matching models.

Training flow models with RL presents several critical challenges: (1) Flow models rely on a deterministic generative process based on ODEs [3], meaning they cannot sample stochastically during inference. In contrast, RL relies on stochastic sampling to explore the environment, learning by trying different actions and improving based on rewards. *This need for stochasticity in RL conflicts with the deterministic nature of flow matching models.* (2) Online RL depends on efficient sampling

[*]Equal contribution, [†]Corresponding author

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

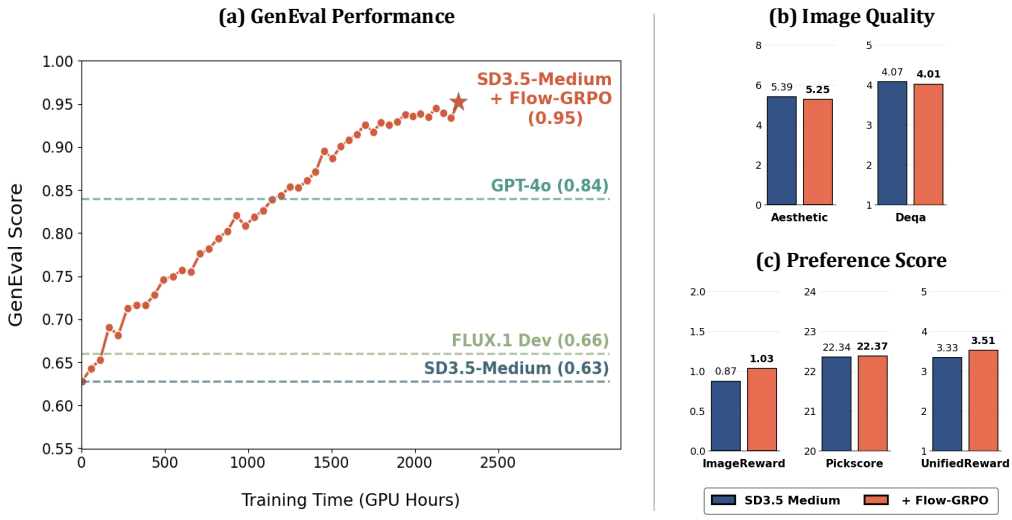


Figure 1: **(a) GenEval performance** rises steadily throughout Flow-GRPO's training and outperforms GPT-4o. **(b) Image quality metrics** on DrawBench [1] remain essentially unchanged. **(c) Human Preference Scores** on DrawBench improves after training. Results show that *Flow-GRPO enhances the desired capability while preserving image quality and exhibiting minimal reward-hacking.*

to collect training data, but flow models typically require many iterative steps to generate each sample, limiting efficiency. This issue is more pronounced with large models [5, 4]. To make RL practical for tasks like image or video generation, *improving sampling efficiency is essential*.

To address these challenges, we propose **Flow-GRPO**, which integrates GRPO [16] into flow matching models for text-to-image (T2I) generation, using two key strategies. First, we adopt the **ODE-to-SDE** strategy to overcome the deterministic nature of the original flow model. By converting the ODE-based flow into an equivalent Stochastic Differential Equation (SDE) framework, we introduce randomness while preserving the original marginal distributions. Second, to improve sampling efficiency in online RL, we apply the **Denoising Reduction** strategy, which reduces denoising steps during training while keeping the full schedule during inference. Our experiments show that using fewer steps maintains performance while significantly reducing data generation costs.

We evaluate Flow-GRPO on T2I tasks with various reward types. (1) Verifiable rewards, using the GenEval [17] benchmark and visual text rendering task. GenEval includes compositional image generation tasks (e.g., generating specific object counts, colors, and spatial relationships), which can be automatically assessed with object detection methods. Flow-GRPO improves the accuracy of Stable Diffusion 3.5 Medium (SD3.5-M) [4] from 63% to 95% on GenEval, outperforming the state-of-the-art GPT-4o [18] model. For visual text rendering, SD3.5-M's accuracy increases from 59% to 92%, greatly enhancing its text generation ability. (2) Model-based rewards, such as the human preference Pickscore [19] reward. These results show that our framework is task independent, demonstrating its generalizability and robustness. Importantly, all improvements are achieved with very little reward hacking, as demonstrated in Figure 1.

To summarize, the contributions of Flow-GRPO are as follows:

- We are the first to introduce GRPO to flow matching models by converting deterministic ODE sampling into SDE sampling, showing the effectiveness of online RL for T2I tasks. Flow-GRPO improves SD3.5-M accuracy from 63% to 95% without noticeably compromising image quality.
- We find that online RL for flow matching models does not require the standard long timesteps for training sample collection. By using fewer denoising steps during training and retaining the original steps during testing, we can significantly accelerate the training process.
- We show that the Kullback-Leibler (KL) constraint effectively prevents reward hacking, where reward increases at the cost of image quality or diversity. KL regularization is not empirically equivalent to early stopping. With a proper KL term, we can match the high reward of the KL-free version while preserving image quality, albeit with longer training.

2 Related Work

RL for LLM. Online RL has effectively improved the reasoning abilities of LLMs, such as DeepSeek-R1 [10] and OpenAI-o1 [11], using policy gradient methods like PPO [20] or value-free GRPO [16]. GRPO is more memory efficient by removing the need for a value network, so we adopt it in this work. PPO can also be applied to flow matching in a similar way.

Diffusion and Flow Matching. Diffusion models [21, 22, 23] add Gaussian noise to data and train a neural network to reverse the process. Sampling uses discrete DDPM steps or probability flow SDE solvers to generate high-fidelity outputs. Flow matching [2, 3] learns a continuous-time normalizing flow by directly matching the velocity field, allowing efficient deterministic sampling with only a few ODE steps. It achieves competitive FID with far fewer denoising steps than diffusion, making it the dominant choice in recent image [4, 5] and video [24, 25, 26, 27] generation models. Recent work [28, 29] unifies diffusion and flow models under an SDE/ODE framework. Our work builds on their theoretical foundations and introduces GRPO to flow-based models.

Alignment for T2I. Recent efforts to align pretrained T2I models with human preferences follow five main directions: (1) direct fine-tuning with differentiable rewards [30, 31, 32, 33]; (2) Reward Weighted Regression (RWR) [34, 35, 36, 37]; (3) Direct Preference Optimization (DPO) and variants [38, 39, 14, 40, 41, 42, 43, 44, 45, 46]; (4) PPO-style policy gradients [47, 48, 49, 50, 51, 52]; (5) training-free alignment methods [53, 54, 55]. These methods have successfully aligned T2I models with human preferences, improving aesthetics and semantic consistency. Building on this progress, we introduce GRPO for flow matching models, the backbone of today's state-of-the-art T2I systems. Concurrent work [56] applies GRPO to text-to-speech flow models, but instead of converting the ODE to an SDE to inject stochasticity, they reformulate velocity prediction by estimating a Gaussian distribution (predicting both the mean and variance of velocity), which requires retraining the pre-trained model. Another study [57] also explores SDE-based stochasticity but focuses on inference-time scaling.

3 Preliminaries

In this section, we introduce the mathematical formulation of flow matching and describe how the denoising process can be mapped as a multi-step MDP.

Flow Matching. Let $\boldsymbol{x}_0 \sim X_0$ be a data sample from the true distribution, and $\boldsymbol{x}_1 \sim X_1$ denote a noise sample. Recent advanced image-generation models (e.g., [4, 5]) and video-generation models (e.g., [24, 26, 25, 27]) adopt the Rectified Flow [3] framework, which defines the "noised" data $\boldsymbol{x}_t$ as

$$\boldsymbol{x}_t = (1-t)\,\boldsymbol{x}_0 \;+\; t\,\boldsymbol{x}_1, \tag{1}$$

for $t \in [0, 1]$. Then a transformer model are trained to directly regress the velocity field $\boldsymbol{v}_\theta(\boldsymbol{x}_t, t)$ by minimizing the Flow Matching objective [2, 3]:

$$\mathcal{L}(\theta) = \mathbb{E}_{t,\,\boldsymbol{x}_0\sim X_0,\,\boldsymbol{x}_1\sim X_1}\big[\,\|\boldsymbol{v} \;-\; \boldsymbol{v}_\theta(\boldsymbol{x}_t, t)\|^2\big], \tag{2}$$

where the target velocity field is $\boldsymbol{v} = \boldsymbol{x}_1 - \boldsymbol{x}_0$.

Denoising as an MDP. As shown in [12], the iterative denoising process in flow matching models can be formulated as a Markov Decision Process (MDP) $(\mathcal{S}, \mathcal{A}, \rho_0, P, R)$. The state at step t is $\boldsymbol{s}_t \triangleq (\boldsymbol{c}, t, \boldsymbol{x}_t)$, the action is the denoised sample $\boldsymbol{a}_t \triangleq \boldsymbol{x}_{t-1}$ predicted by the model, and the policy is $\pi(\boldsymbol{a}_t \mid \boldsymbol{s}_t) \triangleq p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{c})$. The transition is deterministic: $P(\boldsymbol{s}_{t+1} \mid \boldsymbol{s}_t, \boldsymbol{a}_t) \triangleq (\delta_{\boldsymbol{c}}, \delta_{t-1}, \delta_{\boldsymbol{x}_{t-1}})$, and the initial state distribution is $\rho_0(\boldsymbol{s}_0) \triangleq (p(\boldsymbol{c}), \delta_T, \mathcal{N}(\mathbf{0}, \mathbf{I}))$, where δ_y is the Dirac delta distribution centered at y. The reward is only given at the final step: $R(\boldsymbol{s}_t, \boldsymbol{a}_t) \triangleq r(\boldsymbol{x}_0, \boldsymbol{c})$ if $t = 0$, and 0 otherwise.

4 Flow-GRPO

In this section, we present Flow-GRPO, which enhances flow models using online RL. We begin by revisiting the core idea of GRPO [16] and adapting it to flow matching. We then show how to convert

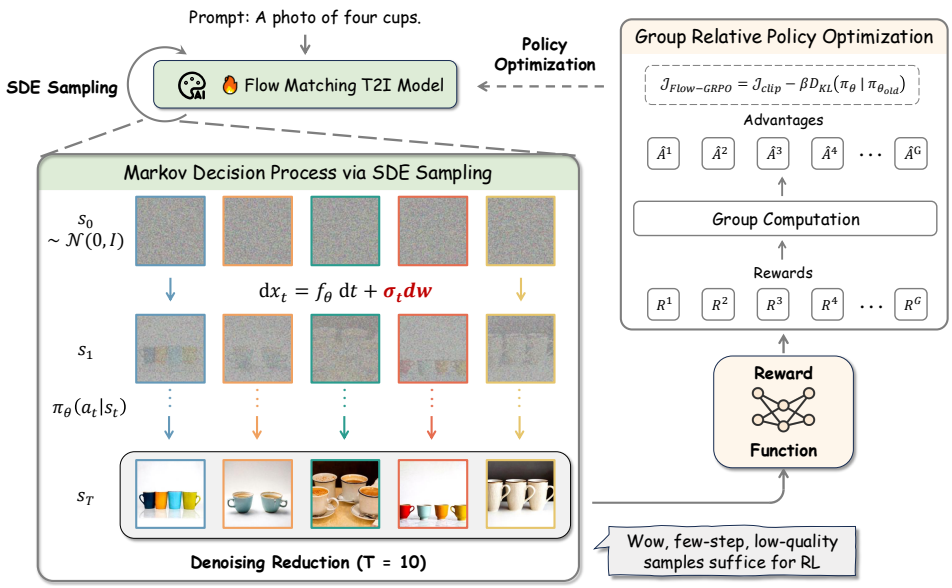


Figure 2: **Overview of Flow-GRPO.** Given a prompt set, we introduce an ODE-to-SDE strategy to enable stochastic sampling for online RL. With Denoising Reduction (only T = 10 steps), we efficiently gather low-quality but still informative trajectories. Rewards from these trajectories feed the GRPO loss, which updates the model online and yields an aligned policy.

the deterministic ODE sampler into a SDE sampler with the same marginal distribution, introducing the stochasticity needed for applying GRPO. Finally, we introduce Denoise Reduction, a practical sampling strategy that significantly speeds up training without sacrificing performance.

GRPO on Flow Matching. RL aims to learn a policy that maximizes the expected cumulative reward. This is often formulated as optimizing a policy π_θ with a regularized objective:

$$\max_\theta \mathbb{E}_{(\boldsymbol{s}_0,\boldsymbol{a}_0,\ldots,\boldsymbol{s}_T,\boldsymbol{a}_T)\sim\pi_\theta}\left[\sum_{t=0}^{T}\left(R(\boldsymbol{s}_t,\boldsymbol{a}_t)-\beta D_{\mathrm{KL}}(\pi_\theta(\cdot\mid\boldsymbol{s}_t)||\pi_{\mathrm{ref}}(\cdot\mid\boldsymbol{s}_t))\right)\right]. \tag{3}$$

Unlike other policy based methods like PPO [20], GRPO [16] provides a lightweight alternative, which introduces a group relative formulation to estimate the advantage.

Recall that the denoising process can be formulated as an MDP, as shown in Section 3. Given a prompt $\boldsymbol{c}$, the flow model p_θ samples a group of G individual images $\{\boldsymbol{x}_0^i\}_{i=1}^G$ and the corresponding reverse-time trajectories $\{(\boldsymbol{x}_T^i,\boldsymbol{x}_{T-1}^i,\cdots,\boldsymbol{x}_0^i)\}_{i=1}^G$. Then, the advantage of the i-th image is calculated by normalizing the group-level rewards as follows:

$$\hat{A}_t^i=\frac{R(\boldsymbol{x}_0^i,\boldsymbol{c})-\mathrm{mean}(\{R(\boldsymbol{x}_0^i,\boldsymbol{c})\}_{i=1}^G)}{\mathrm{std}(\{R(\boldsymbol{x}_0^i,\boldsymbol{c})\}_{i=1}^G)}. \tag{4}$$

GRPO optimizes the policy model by maximizing the following objective:

$$\mathcal{J}_{\text{Flow-GRPO}}(\theta)=\mathbb{E}_{\boldsymbol{c}\sim\mathcal{C},\{\boldsymbol{x}^i\}_{i=1}^G\sim\pi_{\theta_{\text{old}}}(\cdot|\boldsymbol{c})} f(r,\hat{A},\theta,\varepsilon,\beta), \tag{5}$$

where

$$f(r,\hat{A},\theta,\varepsilon,\beta)=\frac{1}{G}\sum_{i=1}^{G}\frac{1}{T}\sum_{t=0}^{T-1}\left(\min\left(r_t^i(\theta)\hat{A}_t^i,\ \mathrm{clip}\Big(r_t^i(\theta),1-\varepsilon,1+\varepsilon\Big)\hat{A}_t^i\right)-\beta D_{\mathrm{KL}}(\pi_\theta||\pi_{\mathrm{ref}})\right),$$

$$r_t^i(\theta)=\frac{p_\theta(\boldsymbol{x}_{t-1}^i\mid\boldsymbol{x}_t^i,\boldsymbol{c})}{p_{\theta_{\text{old}}}(\boldsymbol{x}_{t-1}^i\mid\boldsymbol{x}_t^i,\boldsymbol{c})}.$$

From ODE to SDE. GRPO relies on stochastic sampling in Eq. 4 and Eq. 5 to generate diverse trajectories for advantage estimation and exploration. Diffusion models naturally support this: the

forward process adds Gaussian noise step by step, and the reverse process approximates a score-based SDE solver via a Markov chain with decreasing variance. In contrast, flow matching models use a deterministic ODE for the forward process:

$$\mathrm{d}\boldsymbol{x}_t = \boldsymbol{v}_t \mathrm{d}t, \tag{6}$$

where $\boldsymbol{v}_t$ is learned via the flow matching objective in Eq. 2. A common sampling method is to discretize this ODE, yielding a one-to-one mapping between successive time steps.

This deterministic approach fails to meet the GRPO policy update requirements in two key ways: (1) $r_t^i(\theta)$ in Eq. 5 requires computing $p(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{c})$, which becomes computationally expensive under deterministic dynamics due to divergence estimation. (2) More importantly, RL depends on exploration. As shown in Section 5.3, reduced randomness greatly lowers training efficiency. Deterministic sampling, with no randomness beyond the initial seed, is especially problematic.

To address this limitation, we convert the deterministic Flow-ODE from Eq. 6 into an equivalent SDE that matches the original model's marginal probability density function at all timesteps. We outline the key process here. A detailed proof is provided in Appendix A. Following [23, 28, 29], we construct a reverse-time SDE formulation that preserves the marginal distribution:

$$\mathrm{d}\boldsymbol{x}_t = \left(\boldsymbol{v}_t(\boldsymbol{x}_t) - \frac{\sigma_t^2}{2} \nabla \log p_t(\boldsymbol{x}_t) \right) \mathrm{d}t + \sigma_t \mathrm{d}\boldsymbol{w}, \tag{7}$$

where $\mathrm{d}\boldsymbol{w}$ denotes Wiener process increments and σ_t control the level of stachasticity during generation. For rectified flow, Eq. 7 is specified as:

$$\mathrm{d}\boldsymbol{x}_t = \left[\boldsymbol{v}_t(\boldsymbol{x}_t) + \frac{\sigma_t^2}{2t} \left(\boldsymbol{x}_t + (1-t)\boldsymbol{v}_t(\boldsymbol{x}_t) \right) \right] \mathrm{d}t + \sigma_t \mathrm{d}\boldsymbol{w}. \tag{8}$$

Applying Euler-Maruyama discretization yields the final update rule:

$$\boxed{\boldsymbol{x}_{t+\Delta t} = \boldsymbol{x}_t + \left[\boldsymbol{v}_\theta(\boldsymbol{x}_t, t) + \frac{\sigma_t^2}{2t} \big(\boldsymbol{x}_t + (1-t)\boldsymbol{v}_\theta(\boldsymbol{x}_t, t) \big) \right] \Delta t + \sigma_t \sqrt{\Delta t}\, \epsilon} \tag{9}$$

where $\epsilon \sim \mathcal{N}(0, \boldsymbol{I})$ injects stochasticity. We use $\sigma_t = a\sqrt{\frac{t}{1-t}}$ in this paper, where a is a scalar hyper-parameter that controls the noise level (See Section 5.3 for its impact on performance).

Eq. 9 reveals that the policy $\pi_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{c})$ is an isotropic Gaussian distribution. We can easily compute the KL divergence between π_θ and the reference policy π_{ref} in Eq. 5 as a closed form:

$$D_{\text{KL}}(\pi_\theta || \pi_{\text{ref}}) = \frac{\|\overline{\boldsymbol{x}}_{t+\Delta t,\theta} - \overline{\boldsymbol{x}}_{t+\Delta t,\text{ref}}\|^2}{2\sigma_t^2 \Delta t} = \frac{\Delta t}{2} \left(\frac{\sigma_t(1-t)}{2t} + \frac{1}{\sigma_t} \right)^2 \|\boldsymbol{v}_\theta(\boldsymbol{x}_t, t) - \boldsymbol{v}_{\text{ref}}(\boldsymbol{x}_t, t)\|^2$$

Denoising Reduction. To produce high-quality images, flow models typically require many denoising steps, making data collection costly for online RL. However, we find that large timesteps are unnecessary during online RL training. We can use significantly fewer denoising steps during sample generation, while retaining the original denoising steps during inference to get high-quality samples. Note that we set the timestep T as 10 in training, while the inference timestep T is set as the original default setting ($T = 40$) for SD3.5-M. Our experiments reveals that this approach enables fast training without sacrificing image quality at test time.

5 Experiments

This section empirically evaluates Flow-GRPO's ability to improve flow matching models on three tasks. (1) Composition Image Generation: This task requires precise object arrangement and attribute control. We report the results on GenEval. (2) Visual Text Rendering: a rule-based task that evaluates the accurate rendering of the text specified in the prompt. (3) Human Preference Alignment: This task aims to align T2I models with human preferences.

5.1 Experimental Setup

We introduce three tasks, detailing their respective prompts and reward definitions. For hyperparameter details and compute resource specifications, please refer to Appendix B.3 and Appendix B.4.

Compositional Image Generation. GenEval [17] assesses T2I models on complex compositional prompts—like object counting, spatial relations, and attribute binding—across six difficult compositional image generation tasks. We use its official evaluation pipeline, which detects object bounding boxes and colors, then infers their spatial relations. Training prompts are generated using official GenEval scripts, which apply templates and random combinations to construct the prompt dataset. The test set is strictly deduplicated: prompts differing only in object order (e.g., `"a photo of A and B"` vs. `"a photo of B and A"`) are treated as identical, and these variants are removed from the training set. Based on the base model's initial accuracy across the six tasks, we set the prompt ratio as `Position` : `Counting` : `Attribute Binding` : `Colors` : `Two Objects` : `Single Object` $= 7:5:3:1:1:0$. Rewards are rule-based: (1) **Counting:** $r = 1 - |N_{\text{gen}} - N_{\text{ref}}|/N_{\text{ref}}$; (2) **Position / Color:** If the object count is correct, a partial reward is assigned; the remainder is granted when the predicted position or color is also correct.

Visual Text Rendering [8]. Text is common in images such as posters, book covers, and memes, so the ability to place accurate and coherent text inside the generated images is crucial for T2I models. In our settings, we define an text rendering task, where each prompt follows the template **``**`A sign that says ``text''`. Specifically, the placeholder **``**`text''` is the exact string that should appear in the image. We use GPT4o to produce 20K training prompts and 1K test prompts. Following [58], we measure text fidelity with the reward $r = \max(1 - N_{\text{e}}/N_{\text{ref}}, 0)$, where N_{e} is the minimum edit distance between the rendered text and the target text and N_{ref} is the number of characters inside the quotation marks in the prompt. This reward also serves as our metric of text accuracy.

Human Preference Alignment [19]. This task aims to align T2I models with human preferences. We use `PickScore` [19] as our reward model, which is based on large-scale human annotated pairwise comparisons of images generated from the same prompt. For each image and prompt pair, `PickScore` provides an overall score that evaluates multiple criteria, such as the alignment of the image with the prompt and its visual quality.

Image Quality Evaluation Metric. Since the T2I model is trained to maximize a predefined reward, it is vulnerable to reward hacking, where the reward increases but image quality or diversity declines. This study aims to make online RL effective for T2I generation without noticeably compromising quality or diversity. To detect reward hacking beyond task-specific accuracy, we evaluate four automatic image quality metrics: Aesthetic Score [59], DeQA [60], ImageReward [32], and UnifiedReward [61] (see Appendix B.1 for details). All metrics are computed on DrawBench [1], a comprehensive benchmark with diverse prompts for T2I models.

5.2 Main Results

Figure 1 and Table 1 show Flow-GRPO's GenEval performance steadily improving during training, ultimately outperforming GPT-4o. This occurs while maintaining both image quality metrics and preference scores on DrawBench, a benchmark with diverse and comprehensive prompts for evaluating general model capabilities. Figure 3 offers qualitative comparisons. Beyond Compositional Image Generation, Table 2 details evaluations on Visual Text Rendering and Human Preference tasks. Flow-GRPO improved text rendering ability, again without decreasing image quality metrics and preference scores on DrawBench. See Figures 13, 14 & 15 in Appendix C.6 for related qualitative examples. For the Human Preference task, image quality did not decrease without KL regularization. However, we found that omitting KL caused a collapse in visual diversity, a form of reward hacking discussed further in Section 5.3. These results demonstrate that Flow-GRPO boosts desired capabilities while causing very little degradation to image quality or visual diversity.

Flow-GRPO vs. Other Alignment Methods. We compare Flow-GRPO with several alignment methods: supervised fine-tuning (SFT), Flow-DPO [14, 39], and their online variants. Flow-GRPO consistently outperforms all baselines by a significant margin. At each step, we generate a group of images using the same group size as in Flow-GRPO. The only difference lies in the update rule:

- **SFT**: Select the highest-reward image in each group and fine-tune on it.
- **Flow-DPO**: Use the highest-reward image in each group as the chosen sample and the lowest as the rejected, then apply the DPO loss.

Table 1: **GenEval Result.** Best scores are in blue, second-best in green. Results for models other than SD3.5-M are from [7] or their original papers. Obj.: Object; Attr.: Attribution.

Model	Overall	Single Obj.	Two Obj.	Counting	Colors	Position	Attr. Binding
Diffusion Models							
LDM [62]	0.37	0.92	0.29	0.23	0.70	0.02	0.05
SD1.5 [62]	0.43	0.97	0.38	0.35	0.76	0.04	0.06
SD2.1 [62]	0.50	0.98	0.51	0.44	0.85	0.07	0.17
SD-XL [63]	0.55	0.98	0.74	0.39	0.85	0.15	0.23
DALLE-2 [64]	0.52	0.94	0.66	0.49	0.77	0.10	0.19
DALLE-3 [65]	0.67	0.96	0.87	0.47	0.83	0.43	0.45
Autoregressive Models							
Show-o [66]	0.53	0.95	0.52	0.49	0.82	0.11	0.28
Emu3-Gen [67]	0.54	0.98	0.71	0.34	0.81	0.17	0.21
JanusFlow [68]	0.63	0.97	0.59	0.45	0.83	0.53	0.42
Janus-Pro-7B [69]	0.80	0.99	0.89	0.59	0.90	0.79	0.66
GPT-4o [18]	0.84	0.99	0.92	0.85	0.92	0.75	0.61
Flow Matching Models							
FLUX.1 Dev [5]	0.66	0.98	0.81	0.74	0.79	0.22	0.45
SD3.5-L [4]	0.71	0.98	0.89	0.73	0.83	0.34	0.47
SANA-1.5 4.8B [70]	0.81	0.99	0.93	0.86	0.84	0.59	0.65
SD3.5-M [4]	0.63	0.98	0.78	0.50	0.81	0.24	0.52
SD3.5-M+Flow-GRPO	0.95	1.00	0.99	0.95	0.92	0.99	0.86

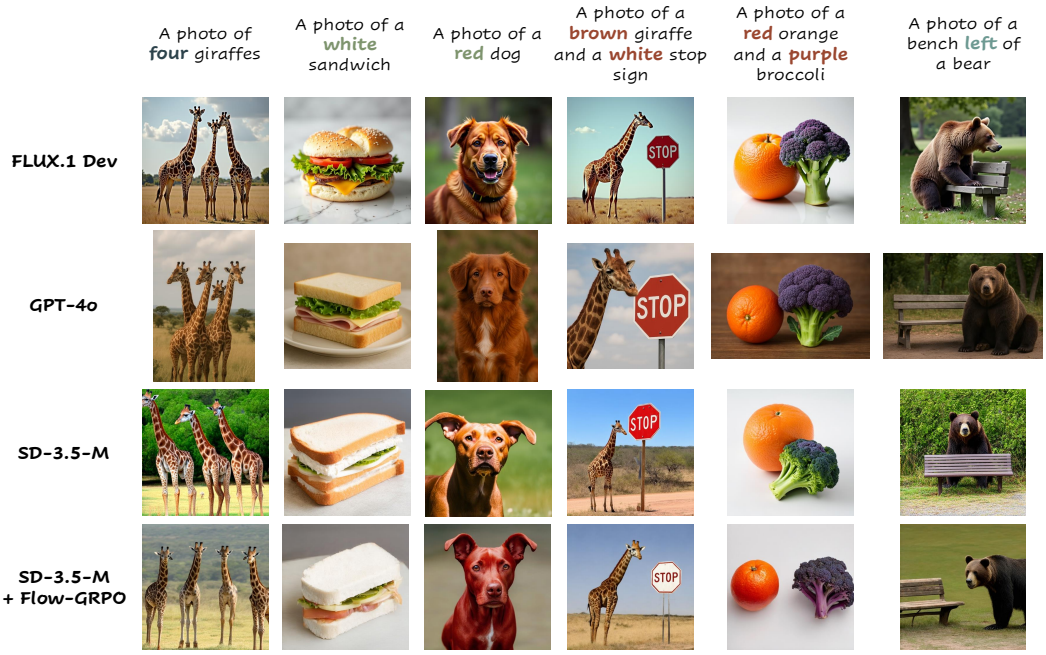


Figure 3: **Qualitative Comparison on the GenEval Benchmark.** Our approach demonstrates superior performance in **Counting**, **Colors**, **Attribute Binding**, **and** **Position**.

Offline variants use a fixed pretrained model for data collection, while online variants update their data collection models every 40 steps. As shown in Figure 4, Flow-GRPO outperforms all baselines. Online DPO also surpasses its offline counterpart, consistent with [15]. For the second-best online DPO, a hyperparameter search on its key parameter β revealed that smaller values are not always optimal; excessively small β values can cause training collapse. Appendix C presents more comprehensive comparisons covering additional methods and tasks.

Table 2: **Performance on Compositional Image Generation, Visual Text Rendering, and Human Preference** benchmarks, evaluated by task performance on test prompts, and by image quality and preference scores on DrawBench prompts. ImgRwd: ImageReward; UniRwd: UnifiedReward.

Model	Task Metric			Image Quality		Preference Score		
	GenEval	OCR Acc.	PickScore	Aesthetic	DeQA	ImgRwd	PickScore	UniRwd
SD3.5-M	0.63	0.59	21.72	5.39	4.07	0.87	22.34	3.33
Compositional Image Generation								
Flow-GRPO (w/o KL)	0.95	—	—	4.93	2.77	0.44	21.16	2.94
Flow-GRPO (w/ KL)	0.95	—	—	5.25	4.01	1.03	22.37	3.51
Visual Text Rendering								
Flow-GRPO (w/o KL)	—	0.93	—	5.13	3.66	0.58	21.79	3.15
Flow-GRPO (w/ KL)	—	0.92	—	5.32	4.06	0.95	22.44	3.42
Human Preference Alignment								
Flow-GRPO (w/o KL)	—	—	23.41	6.15	4.16	1.24	23.56	3.57
Flow-GRPO (w/ KL)	—	—	23.31	5.92	4.22	1.28	23.53	3.66

Figure 4: Comparison with Other Alignment Methods on the Compositional Generation Task.

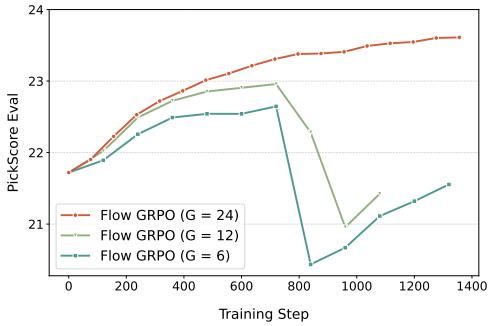


Figure 5: Ablation Studies on Different Group Size G. Higher group size performs better.

5.3 Analysis

This section presents several analyses to better understand the behavior and robustness of Flow-GRPO. We examine issues such as reward hacking, the impact of denoising reduction and noise levels, the effect of group size, and the model's generalization ability. We provide additional analyses in the Appendix C.

Reward Hacking. We use KL regularization to mitigate reward hacking by tuning the KL coefficient to keep the divergence small and nearly constant during training, keeping the model close to its pretrained weights. This allows task-specific reward optimization without harming overall performance. As shown in Table 2, removing the KL constraint for Compositional Image Generation and Visual Text Rendering significantly reduces image quality and preference scores on DrawBench. In contrast, a properly tuned KL preserves quality while achieving similar gains on task-specific metrics. In the Human Preference Alignment task, removing KL does not affect image quality, likely due to overlap between PickScore and evaluation metrics, but causes a collapse in visual diversity. Outputs converge to a single style, with different seeds producing nearly identical results. KL regularization prevents this collapse and maintains diversity. See Figure 12 in Appendix C.5 for training curves and Figure 6 for more examples.

Effect of Denoising Reduction. Figure 7 (a) highlights Denoising Reduction's significant impact on accelerating training. To explore how different timesteps affect optimization, these experiments are conducted without the KL constraint. Reducing data collection timesteps from 40 to 10 achieves over a $4\times$ speedup across all three tasks, without impacting final reward. Further reducing to 5 does not consistently improve speed and sometimes slows training, so we choose 10 timesteps for later

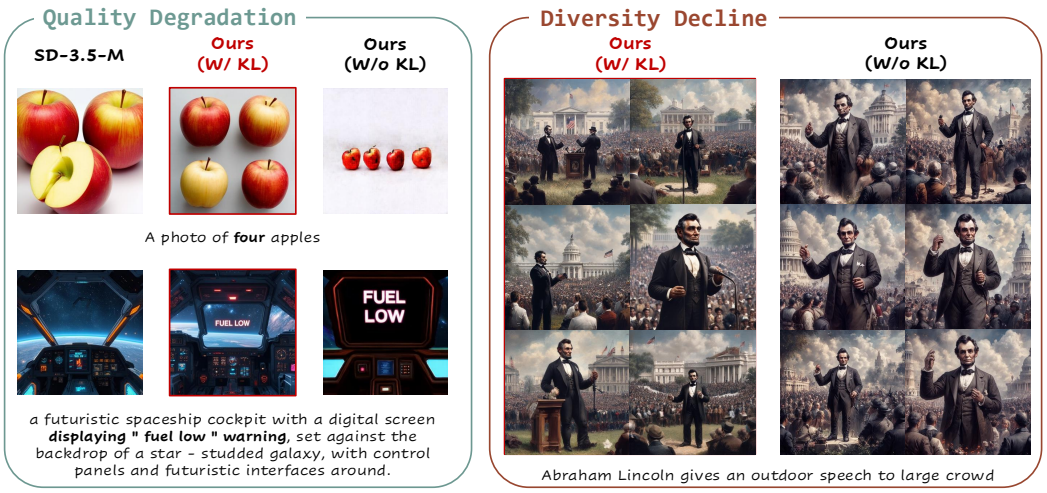
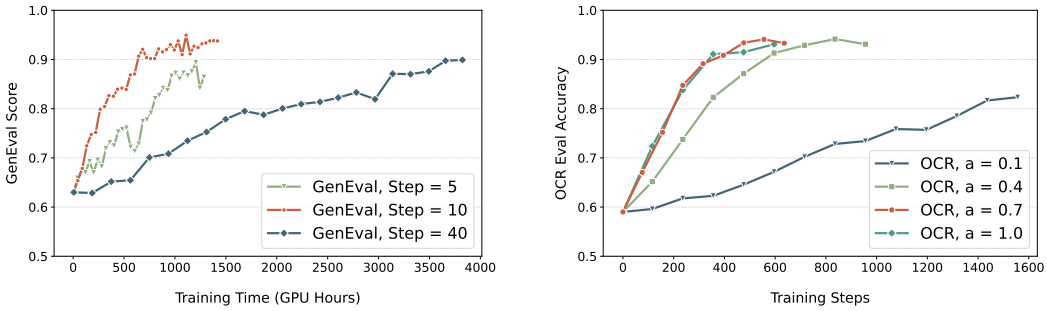


Figure 6: **Effect of KL Regularization.** The KL penalty effectively suppresses reward hacking, preventing **Quality Degradation** (for GenEval and OCR) and **Diversity Decline** (for PickScore).

experiments. For the other two tasks, learning curves of reward versus training time are presented in Figure 9 in the Appendix C.2.

Effect of Noise Level. Higher σ_t in the SDE boosts image diversity and exploration, vital for RL training. We control this exploration with a noise level a (Eq. 9). Figure 7 (b) shows the impact of a on performance. A small a (e.g., 0.1) limits exploration and slows reward improvement. Increasing a (up to 0.7) boosts exploration and speeds up reward gains. Beyond this point (e.g., from 0.7 to 1.0), further increases provide no additional benefit, as exploration is already sufficient. We also observe that injecting too much noise by further increasing a degrades image quality, resulting in zero reward and failed training.



(a). Effect of Denoising Reduction on GenEval

(b). Effect of Noise Level Ablation on the OCR

Figure 7: Ablation studies on our critical design choices. **(a) Denoising Reduction**: Fewer denoising steps accelerate convergence and yield similar performance. **(b) Noise Level:** Moderate noise level ($a = 0.7$) maximises OCR accuracy, while too little noise hampers exploration.

Effect of Group Size. Figure 5 shows the effect of group size G using PickScore as the reward function. When the group size was reduced to $G = 12$ and $G = 6$, training became unstable and eventually collapsed, whereas $G = 24$ remained stable throughout the process. We observe that smaller group sizes produce inaccurate advantage estimates, increasing variance and leading to training collapse, a phenomenon also reported in [71, 72].

Generalization Analysis. Flow-GRPO demonstrates strong generalization on unseen scenarios from GenEval (Table 4). Specifically, it captures object number, color, and spatial relations, generalizing well to unseen object classes. It also effectively controls object count, generalizing from training on $2 - 4$ objects to generate $5 - 6$ or 12 objects. Furthermore, Table 3 shows Flow-GRPO achieves

significant gains on T2I-CompBench++ [6, 73]. This comprehensive benchmark for open-world compositional T2I generation features object classes and relationships substantially different from our model's GenEval-style training data.

Table 3: **T2I-CompBench++ Result.** This evaluation uses the same model presented in Table 1, which was trained on the GenEval-generated dataset. The best score is in blue.

Model	Color	Shape	Texture	2D-Spatial	3D-Spatial	Numeracy	Non-Spatial
Janus-Pro-7B [69]	0.5145	0.3323	0.4069	0.1566	0.2753	0.4406	0.3137
EMU3 [67]	0.7913	0.5846	0.7422	—	—	—	—
FLUX.1 Dev [5]	0.7407	0.5718	0.6922	0.2863	0.3866	0.6185	0.3127
SD3.5-M [4]	0.7994	0.5669	0.7338	0.2850	0.3739	0.5927	0.3146
SD3.5-M+Flow-GRPO	0.8379	0.6130	0.7236	0.5447	0.4471	0.6752	0.3195

Table 4: **Flow-GRPO demonstrates strong generalization.** Unseen Objects: Trained on 60 object classes, evaluated on 20 unseen classes. Unseen Counting: Trained to render 2, 3, or 4 objects, and evaluated in two settings: rendering 5 or 6 objects, and rendering 12 objects.

Method	Unseen Objects							Unseen Counting	
	Overall	Single Obj.	Two Obj.	Counting	Colors	Position	Attr. Binding	5–6 Objects	12 Objects
SD3.5-M	0.64	0.96	0.73	0.53	0.87	0.26	0.47	0.13	0.02
SD3.5-M+Flow-GRPO	0.90	1.00	0.94	0.86	0.97	0.84	0.77	0.48	0.12

6 Conclusion

We have presented Flow-GRPO, the first method to integrate online policy gradient RL into flow matching models. By converting deterministic ODEs to SDEs and reducing denoising steps during training, Flow-GRPO enables efficient RL-based optimization without noticeably compromising image quality or diversity. Our method significantly improves performance on compositional generation, text rendering, and human preference alignment, with minimal reward hacking. Flow-GRPO offers a simple and general framework for applying online RL to flow-based generative models.

Limitations & Future Work. Although this work focuses on T2I tasks, Flow-GRPO has potential for video generation [25, 27], raising several future directions: (1) Reward Design: Simple heuristics, such as using object detectors or trackers as rule-based rewards, can encourage physical realism and temporal consistency, but more advanced reward models are needed. (2) Balancing Multiple Rewards: Video generation requires optimizing multiple objectives, including realism, smoothness, and coherence. Balancing these competing goals remains challenging and demands careful tuning. (3) Scalability: Video generation is far more resource-intensive than T2I, so applying Flow-GRPO at scale requires more efficient data collection and training pipelines. Additionally, better methods for preventing reward hacking are worth exploring. While KL regularization helps significantly, it requires longer training and occasional reward hacking occurs for certain prompts.

Acknowledgements

This work was partially supported by the JC STEM Lab of AI for Science and Engineering, funded by The Hong Kong Jockey Club Charities Trust, the Research Grants Council of Hong Kong (Project No. CUHK14213224). We gratefully acknowledge Mingwu Zheng for his insightful discussions on the proof and Zhanhui Zhou for his valuable comments that improved the clarity of this paper.

References

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

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

[3] 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.

[4] 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.

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

[6] Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2i-compbench: A comprehensive benchmark for open-world compositional text-to-image generation. *Advances in Neural Information Processing Systems*, 36:78723–78747, 2023.

[7] Zhiyuan Yan, Junyan Ye, Weijia Li, Zilong Huang, Shenghai Yuan, Xiangyang He, Kaiqing Lin, Jun He, Conghui He, and Li Yuan. Gpt-imgeval: A comprehensive benchmark for diagnosing gpt4o in image generation. *arXiv preprint arXiv:2504.02782*, 2025.

[8] Jingye Chen, Yupan Huang, Tengchao Lv, Lei Cui, Qifeng Chen, and Furu Wei. Textdiffuser: Diffusion models as text painters. *Advances in Neural Information Processing Systems*, 36:9353–9387, 2023.

[9] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

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

[11] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

[12] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*, 2023.

[13] 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.

[14] Jie Liu, Gongye Liu, Jiajun Liang, Ziyang Yuan, Xiaokun Liu, Mingwu Zheng, Xiele Wu, Qiulin Wang, Wenyu Qin, Menghan Xia, et al. Improving video generation with human feedback. *arXiv preprint arXiv:2501.13918*, 2025.

[15] Guibin Chen, Dixuan Lin, Jiangping Yang, Chunze Lin, Juncheng Zhu, Mingyuan Fan, Hao Zhang, Sheng Chen, Zheng Chen, Chengchen Ma, et al. Skyreels-v2: Infinite-length film generative model. *arXiv preprint arXiv:2504.13074*, 2025.

[16] 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.

[17] Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36:52132–52152, 2023.

[18] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[19] 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.

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

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

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

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

[24] Kuaishou. Kling ai. *https://klingai.kuaishou.com/*, 2024.

[25] Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[26] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. Video generation models as world simulators. *OpenAI Blog*, 1:8, 2024.

[27] Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

[28] Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

[29] Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky TQ Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. *arXiv preprint arXiv:2409.08861*, 2024.

[30] Mihir Prabhudesai, Anirudh Goyal, Deepak Pathak, and Katerina Fragkiadaki. Aligning text-to-image diffusion models with reward backpropagation. *arXiv preprint arXiv:2310.03739*, 2023.

[31] Kevin Clark, Paul Vicol, Kevin Swersky, and David J Fleet. Directly fine-tuning diffusion models on differentiable rewards. *arXiv preprint arXiv:2309.17400*, 2023.

[32] 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, 2024.

[33] Mihir Prabhudesai, Russell Mendonca, Zheyang Qin, Katerina Fragkiadaki, and Deepak Pathak. Video diffusion alignment via reward gradients. *arXiv preprint arXiv:2407.08737*, 2024.

[34] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

[35] Jiajun Fan, Shuaike Shen, Chaoran Cheng, Yuxin Chen, Chumeng Liang, and Ge Liu. Online reward-weighted fine-tuning of flow matching with wasserstein regularization. In *The Thirteenth International Conference on Learning Representations*, 2025.

[36] Kimin Lee, Hao Liu, Moonkyung Ryu, Olivia Watkins, Yuqing Du, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, and Shixiang Shane Gu. Aligning text-to-image models using human feedback. *arXiv preprint arXiv:2302.12192*, 2023.

[37] Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*, 2023.

[38] 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, 2024.

[39] 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.

[40] Kai Yang, Jian Tao, Jiafei Lyu, Chunjiang Ge, Jiaxin Chen, Weihan Shen, Xiaolong Zhu, and Xiu Li. Using human feedback to fine-tune diffusion models without any reward model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8941–8951, 2024.

[41] Zhanhao Liang, Yuhui Yuan, Shuyang Gu, Bohan Chen, Tiankai Hang, Ji Li, and Liang Zheng. Step-aware preference optimization: Aligning preference with denoising performance at each step. *arXiv preprint arXiv:2406.04314*, 2024.

[42] Huizhuo Yuan, Zixiang Chen, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning of diffusion models for text-to-image generation. *arXiv preprint arXiv:2402.10210*, 2024.

[43] Runtao Liu, Haoyu Wu, Zheng Ziqiang, Chen Wei, Yingqing He, Renjie Pi, and Qifeng Chen. Videodpo: Omni-preference alignment for video diffusion generation. *arXiv preprint arXiv:2412.14167*, 2024.

[44] Jiacheng Zhang, Jie Wu, Weifeng Chen, Yatai Ji, Xuefeng Xiao, Weilin Huang, and Kai Han. Onlinevpo: Align video diffusion model with online video-centric preference optimization. *arXiv preprint arXiv:2412.15159*, 2024.

[45] Hiroki Furuta, Heiga Zen, Dale Schuurmans, Aleksandra Faust, Yutaka Matsuo, Percy Liang, and Sherry Yang. Improving dynamic object interactions in text-to-video generation with ai feedback. *arXiv preprint arXiv:2412.02617*, 2024.

[46] Zhanhao Liang, Yuhui Yuan, Shuyang Gu, Bohan Chen, Tiankai Hang, Mingxi Cheng, Ji Li, and Liang Zheng. Aesthetic post-training diffusion models from generic preferences with step-by-step preference optimization. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 13199–13208, 2025.

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

[48] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*, 2023.

[49] 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. *Advances in Neural Information Processing Systems*, 36, 2024.

[50] Shashank Gupta, Chaitanya Ahuja, Tsung-Yu Lin, Sreya Dutta Roy, Harrie Oosterhuis, Maarten de Rijke, and Satya Narayan Shukla. A simple and effective reinforcement learning method for text-to-image diffusion fine-tuning. *arXiv preprint arXiv:2503.00897*, 2025.

[51] Zichen Miao, Jiang Wang, Ze Wang, Zhengyuan Yang, Lijuan Wang, Qiang Qiu, and Zicheng Liu. Training diffusion models towards diverse image generation with reinforcement learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10844–10853, 2024.

[52] Hanyang Zhao, Haoxian Chen, Ji Zhang, David D Yao, and Wenpin Tang. Score as action: Fine-tuning diffusion generative models by continuous-time reinforcement learning. *arXiv preprint arXiv:2502.01819*, 2025.

[53] Po-Hung Yeh, Kuang-Huei Lee, and Jun-Cheng Chen. Training-free diffusion model alignment with sampling demons. *arXiv preprint arXiv:2410.05760*, 2024.

[54] Zhiwei Tang, Jiangweizhi Peng, Jiasheng Tang, Mingyi Hong, Fan Wang, and Tsung-Hui Chang. Tuning-free alignment of diffusion models with direct noise optimization. *arXiv preprint arXiv:2405.18881*, 2024.

[55] Jiaming Song, Qinsheng Zhang, Hongxu Yin, Morteza Mardani, Ming-Yu Liu, Jan Kautz, Yongxin Chen, and Arash Vahdat. Loss-guided diffusion models for plug-and-play controllable generation. In *International Conference on Machine Learning*, pages 32483–32498. PMLR, 2023.

[56] Xiaohui Sun, Ruitong Xiao, Jianye Mo, Bowen Wu, Qun Yu, and Baoxun Wang. F5r-tts: Improving flow matching based text-to-speech with group relative policy optimization. *arXiv preprint arXiv:2504.02407*, 2025.

[57] Jaihoon Kim, Taehoon Yoon, Jisung Hwang, and Minhyuk Sung. Inference-time scaling for flow models via stochastic generation and rollover budget forcing. *arXiv preprint arXiv:2503.19385*, 2025.

[58] Lixue Gong, Xiaoxia Hou, Fanshi Li, Liang Li, Xiaochen Lian, Fei Liu, Liyang Liu, Wei Liu, Wei Lu, Yichun Shi, et al. Seedream 2.0: A native chinese-english bilingual image generation foundation model. *arXiv preprint arXiv:2503.07703*, 2025.

[59] Chrisoph Schuhmann. Laion aesthetics, Aug 2022.

[60] Zhiyuan You, Xin Cai, Jinjin Gu, Tianfan Xue, and Chao Dong. Teaching large language models to regress accurate image quality scores using score distribution. *arXiv preprint arXiv:2501.11561*, 2025.

[61] Yibin Wang, Yuhang Zang, Hao Li, Cheng Jin, and Jiaqi Wang. Unified reward model for multimodal understanding and generation. *arXiv preprint arXiv:2503.05236*, 2025.

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

[63] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[64] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

[65] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023.

[66] Jinheng Xie, Weijia Mao, Zechen Bai, David Junhao Zhang, Weihao Wang, Kevin Qinghong Lin, Yuchao Gu, Zhijie Chen, Zhenheng Yang, and Mike Zheng Shou. Show-o: One single transformer to unify multimodal understanding and generation. *arXiv preprint arXiv:2408.12528*, 2024.

[67] Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, et al. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*, 2024.

[68] Yiyang Ma, Xingchao Liu, Xiaokang Chen, Wen Liu, Chengyue Wu, Zhiyu Wu, Zizheng Pan, Zhenda Xie, Haowei Zhang, Liang Zhao, et al. Janusflow: Harmonizing autoregression and rectified flow for unified multimodal understanding and generation. *arXiv preprint arXiv:2411.07975*, 2024.

[69] Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-pro: Unified multimodal understanding and generation with data and model scaling. *arXiv preprint arXiv:2501.17811*, 2025.

[70] Enze Xie, Junsong Chen, Yuyang Zhao, Jincheng Yu, Ligeng Zhu, Yujun Lin, Zhekai Zhang, Muyang Li, Junyu Chen, Han Cai, et al. Sana 1.5: Efficient scaling of training-time and inference-time compute in linear diffusion transformer. *arXiv preprint arXiv:2501.18427*, 2025.

[71] Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models. *arXiv preprint arXiv:2505.24864*, 2025.

[72] Yang Chen, Zhuolin Yang, Zihan Liu, Chankyu Lee, Peng Xu, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. Acereason-nemotron: Advancing math and code reasoning through reinforcement learning. *arXiv preprint arXiv:2505.16400*, 2025.

[73] Kaiyi Huang, Chengqi Duan, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2i-compbench++: An enhanced and comprehensive benchmark for compositional text-to-image generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[74] Bernt Øksendal and Bernt Øksendal. *Stochastic differential equations*. Springer, 2003.

[75] Brian DO Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

[76] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Appendix of Flow-GRPO: Training Flow Matching Models via Online RL

Our Appendix consists of 4 sections. Readers can click on each section number to navigate to the corresponding section:

- Section A provides detailed derivations of stochastic sampling in flow matching models.
- Section B presents details about our experimental setup.
- Section C offers some additional experimental results, including 1) the comparison with other alignment methods, 2) ablation of denoising reduction on OCR accuracy and pickscore, 3) ablation of initial noise, 4) additional results on FLUX.1-Dev, 5) the learning curves of Flow-GRPO on three tasks, 6) additional qualitative results, and 7) evolution of evaluation images during training.
- Section D provides a visualization of training samples under the denoising reduction strategy.

In addition to this Appendix, we also provide more visualization results, see this website. We encourage the readers to consult this HTML page for a more intuitive assessment of the improvements brought by Flow-GRPO.

A Mathematical Derivations for Stochastic Sampling using Flow Models

We present a detailed proof here. To compute $p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{c})$ in Equation 5 during forward sampling, we adapt flow models to a stochastic differential equation (SDE). While flow models normally follow a deterministic ODE:

$$\mathrm{d}\boldsymbol{x}_t = \boldsymbol{v}_t \mathrm{d}t \tag{10}$$

We consider its stochastic counterpart. Inspired by the derivation from SDE to its probability flow ODE in SGMs [23], we aim to construct a forward SDE with specific drift and diffusion coefficients so that its marginal distribution matches that of Eq. 10. We begin with the generic form of SDE:

$$\mathrm{d}\boldsymbol{x}_t = f_{\text{SDE}}(\boldsymbol{x}_t, t)\mathrm{d}t + \sigma_t \mathrm{d}\boldsymbol{w}, \tag{11}$$

Its marginal probability density $p_t(\boldsymbol{x})$ evolves according to the Fokker–Planck equation [74], i.e.,

$$\partial_t p_t(x) = -\nabla \cdot [f_{\text{SDE}}(\boldsymbol{x}_t, t) p_t(\boldsymbol{x})] + \frac{1}{2}\nabla^2[\sigma_t^2 p_t(\boldsymbol{x})] \tag{12}$$

Similarly, the marginal probability density associated with Eq. 10 evolves:

$$\partial_t p_t(\boldsymbol{x}) = -\nabla \cdot [\boldsymbol{v}_t(\boldsymbol{x}_t, t) p_t(\boldsymbol{x})] \tag{13}$$

To ensure that the stochastic process shares the same marginal distribution as the ODE, we impose:

$$-\nabla \cdot [f_{\text{SDE}}\, p_t(\boldsymbol{x})] + \frac{1}{2}\nabla^2[\sigma_t^2 p_t(\boldsymbol{x})] = -\nabla \cdot [\boldsymbol{v}_t(\boldsymbol{x}_t, t) p_t(\boldsymbol{x})] \tag{14}$$

Observing that

$$\begin{aligned} \nabla^2[\sigma_t^2 p_t(\boldsymbol{x})] &= \sigma_t^2 \nabla^2 p_t(\boldsymbol{x}) \\ &= \sigma_t^2 \nabla \cdot (\nabla p_t(\boldsymbol{x})) \\ &= \sigma_t^2 \nabla \cdot (p_t(\boldsymbol{x}) \nabla \log p_t(\boldsymbol{x})) \end{aligned} \tag{15}$$

Substituting Eq. 15 to Eq. 14, we arrive at the drift coefficients of the target forward SDE:

$$f_{\text{SDE}} = \boldsymbol{v}_t(\boldsymbol{x}_t, t) + \frac{\sigma_t^2}{2} \nabla \log p_t(\boldsymbol{x}) \tag{16}$$

Hence, we can rewrite the forward SDE in Eq. 11 as:

$$\mathrm{d}\boldsymbol{x}_t = \left(\boldsymbol{v}_t(\boldsymbol{x}_t) + \frac{\sigma_t^2}{2} \nabla \log p_t(\boldsymbol{x}_t) \right) \mathrm{d}t + \sigma_t \mathrm{d}\boldsymbol{w}, \tag{17}$$

where $\mathrm{d}\boldsymbol{w}$ denotes Wiener process increments, and σ_t is the diffusion coefficient controlling the level of stochasticity during sampling.

The relationship between forward and reverse-time SDEs has been established in [75, 23]. Specifically, if the forward SDE takes the form

$$\mathrm{d}\boldsymbol{x}_t = f(\boldsymbol{x}_t, t)\, \mathrm{d}t + g(t)\, \mathrm{d}\boldsymbol{w}, \tag{18}$$

then the corresponding reverse-time SDE is

$$\mathrm{d}\boldsymbol{x}_t = \left[f(\boldsymbol{x}_t, t) - g^2(t) \nabla \log p_t(\boldsymbol{x}_t) \right]\, \mathrm{d}t + g(t)\, \mathrm{d}\overline{\boldsymbol{w}}. \tag{19}$$

Setting $g(t) = \sigma_t$, we obtain the reverse-time SDE corresponding to Eq. 17 as

$$\mathrm{d}\boldsymbol{x}_t = \left[\boldsymbol{v}_t(\boldsymbol{x}_t) + \frac{\sigma_t^2}{2} \nabla \log p_t(\boldsymbol{x}_t) - \sigma_t^2 \nabla \log p_t(\boldsymbol{x}_t) \right] \mathrm{d}t + \sigma_t \mathrm{d}\overline{\boldsymbol{w}}. \tag{20}$$

We thus arrive at the final form of the reverse-time SDE:

$$\boxed{\mathrm{d}\boldsymbol{x}_t = \left(\boldsymbol{v}_t(\boldsymbol{x}_t) - \frac{\sigma_t^2}{2} \nabla \log p_t(\boldsymbol{x}_t) \right) \mathrm{d}t + \sigma_t \mathrm{d}\boldsymbol{w},} \tag{21}$$

Once the score function $\nabla \log p_t(\boldsymbol{x}_t)$ is available, the process can be simulated directly. For flow matching, this score is implicitly linked to the velocity field $\boldsymbol{v}_t$.

Specifically, let $\dot{\alpha}_t \equiv \partial \alpha_t / \partial t$. All expectations are over $\boldsymbol{x}_0 \sim X_0$ and $\boldsymbol{x}_1 \sim \mathcal{N}(0, \boldsymbol{I})$, where X_0 is the data distribution.

For the linear interpolation $\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \beta_t \boldsymbol{x}_1$, we have:

$$p_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t \mid \alpha_t \boldsymbol{x}_0, \beta_t^2 \boldsymbol{I}\right), \tag{22}$$

yielding the conditional score:

$$\nabla \log p_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0) = -\frac{\boldsymbol{x}_t - \alpha_t \boldsymbol{x}_0}{\beta_t^2} = -\frac{\boldsymbol{x}_1}{\beta_t}. \tag{23}$$

The marginal score becomes:

$$\begin{aligned} \nabla \log p_t(\boldsymbol{x}_t) &= \mathbb{E}\left[\nabla \log p_{t|0}(\boldsymbol{x}_t|\boldsymbol{x}_0) \mid \boldsymbol{x}_t\right] \\ &= -\frac{1}{\beta_t}\mathbb{E}[\boldsymbol{x}_1 \mid \boldsymbol{x}_t]. \end{aligned} \tag{24}$$

For the velocity field $\boldsymbol{v}_t(\boldsymbol{x}_t)$, we derive:

$$\begin{aligned} \boldsymbol{v}_t(\boldsymbol{x}) &= \mathbb{E}\left[\dot{\alpha}_t \boldsymbol{x}_0 + \dot{\beta}_t \boldsymbol{x}_1 \mid \boldsymbol{x}_t = \boldsymbol{x}\right] \\ &= \dot{\alpha}_t \mathbb{E}[\boldsymbol{x}_0 \mid \boldsymbol{x}_t = \boldsymbol{x}] + \dot{\beta}_t \mathbb{E}[\boldsymbol{x}_1 \mid \boldsymbol{x}_t = \boldsymbol{x}] \\ &= \dot{\alpha}_t \mathbb{E}\left[\frac{\boldsymbol{x}_t - \beta_t \boldsymbol{x}_1}{\alpha_t} \mid \boldsymbol{x}_t = \boldsymbol{x}\right] + \dot{\beta}_t \mathbb{E}[\boldsymbol{x}_1 \mid \boldsymbol{x}_t = \boldsymbol{x}] \\ &= \frac{\dot{\alpha}_t}{\alpha_t}\boldsymbol{x} - \frac{\dot{\alpha}_t \beta_t}{\alpha_t}\mathbb{E}[\boldsymbol{x}_1 \mid \boldsymbol{x}_t = \boldsymbol{x}] + \dot{\beta}_t \mathbb{E}[\boldsymbol{x}_1 \mid \boldsymbol{x}_t = \boldsymbol{x}] \\ &= \frac{\dot{\alpha}_t}{\alpha_t}\boldsymbol{x} - \left(\dot{\beta}_t \beta_t - \frac{\dot{\alpha}_t \beta_t^2}{\alpha_t}\right) \nabla \log p_t(\boldsymbol{x}), \end{aligned} \tag{25}$$

Substituting $\alpha_t = 1 - t$ and $\beta_t = t$ simplifies Equation 25 to:

$$\boldsymbol{v}_t(\boldsymbol{x}) = -\frac{\boldsymbol{x}}{1-t} - \frac{t}{1-t}\nabla \log p_t(\boldsymbol{x}). \tag{26}$$

Solving for the score yields:

$$\nabla \log p_t(\boldsymbol{x}) = -\frac{\boldsymbol{x}}{t} - \frac{1-t}{t}\boldsymbol{v}_t(\boldsymbol{x}). \tag{27}$$

Substituting Equation 27 into 21 gives the final SDE:

$$\mathrm{d}\boldsymbol{x}_t = \left[\boldsymbol{v}_t(\boldsymbol{x}_t) + \frac{\sigma_t^2}{2t}\left(\boldsymbol{x}_t + (1-t)\boldsymbol{v}_t(\boldsymbol{x}_t)\right)\right]\mathrm{d}t + \sigma_t \mathrm{d}\boldsymbol{w}. \tag{28}$$

Applying Euler-Maruyama discretization yields the update rule:

$$\boxed{\boldsymbol{x}_{t+\Delta t} = \boldsymbol{x}_t + \left[\boldsymbol{v}_\theta(\boldsymbol{x}_t, t) + \frac{\sigma_t^2}{2t}\left(\boldsymbol{x}_t + (1-t)\boldsymbol{v}_\theta(\boldsymbol{x}_t, t)\right)\right]\Delta t + \sigma_t \sqrt{\Delta t}\,\epsilon,} \tag{29}$$

where $\epsilon \sim \mathcal{N}(0, \boldsymbol{I})$ injects stochasticity.

B Further Details on the Experimental Setup

B.1 Quality Metrics

The details of quality metrics are as follows:

- Aesthetic score [59]: a CLIP-based linear regressor that predicts an image's aesthetic score.
- DeQA score [60]: a multimodal large language model based image-quality assessment (IQA) model that quantifies how distortions, texture damage, and other low-level artefacts affect perceived quality.
- ImageReward [32]: a general purpose T2I human preference reward model that captures text–image alignment, visual fidelity, and harmlessness.
- UnifiedReward [61]: a recently proposed unified reward model for multimodal understanding and generation that currently achieves state-of-the-art performance on the human preference assessment leaderboard.

B.2 Model Specification

The following table lists the base model and the reward models and their corresponding links.

Models	Links
`SD3.5-M` [4]	https://huggingface.co/stabilityai/stable-diffusion-3.5-medium
`Aesthetic Score` [59]	https://github.com/LAION-AI/aesthetic-predictor
`PickScore` [19]	https://huggingface.co/yuvalkirstain/PickScore_v1
`DeQA score` [60]	https://huggingface.co/zhiyuanyou/DeQA-Score-Mix3
`ImageReward` [32]	https://huggingface.co/THUDM/ImageReward
`UnifiedReward` [61]	https://huggingface.co/CodeGoat24/UnifiedReward-7b-v1.5

B.3 Hyperparameters Specification

Except for β, GRPO hyperparameters are fixed across tasks. We use a sampling timestep $T = 10$ and an evaluation timestep $T = 40$. Other settings include a group size $G = 24$, an noise level $a = 0.7$ and an image resolution of 512. The KL ratio β is set to 0.04 for GenEval and Text Rendering, and 0.01 for Pickscore. We use Lora with $\alpha = 64$ and $r = 32$.

B.4 Compute Resources Specification

We train our model using 24 NVIDIA A800 GPUs. The learning curves in Appendix C.5 provide details on the specific GPU hours.

C Extended Experimental Results

C.1 Flow-GRPO vs. Other Alignment Methods

We compare Flow-GRPO with several alignment methods: supervised fine-tuning (SFT), reward-weighted regression (Flow-RWR [14, 76]), Flow-DPO [14], and their online variants. Flow-GRPO consistently outperforms all baselines by a significant margin. At each step, we generate a group of images using the same group size as in Flow-GRPO. The only difference lies in the update rule:

- **SFT**: Select the highest-reward image in each group and fine-tune on it.
- **Flow-RWR** [14, 76]: Apply a softmax over rewards in each group and perform reward-weighted likelihood maximization.
- **Flow-DPO** [14, 39]: Use the highest-reward image in each group as the chosen sample and the lowest as the rejected, then apply the DPO loss.

Offline variants use a fixed pretrained model for data collection, while online variants update their data collection model every 40 steps. As shown in Figure 8, Flow-GRPO outperforms all other methods. The figure also indicates that DPO and SFT improve over time. In contrast, RWR does not, which aligns with experimental findings on RWR in [12]. Additionally, Online DPO surpasses offline DPO, aligning with [15]'s finding that online DPO performs better. For the second-best online DPO, a hyperparameter search on its key parameter β revealed that smaller values are not always optimal; excessively small β values can cause training collapse.

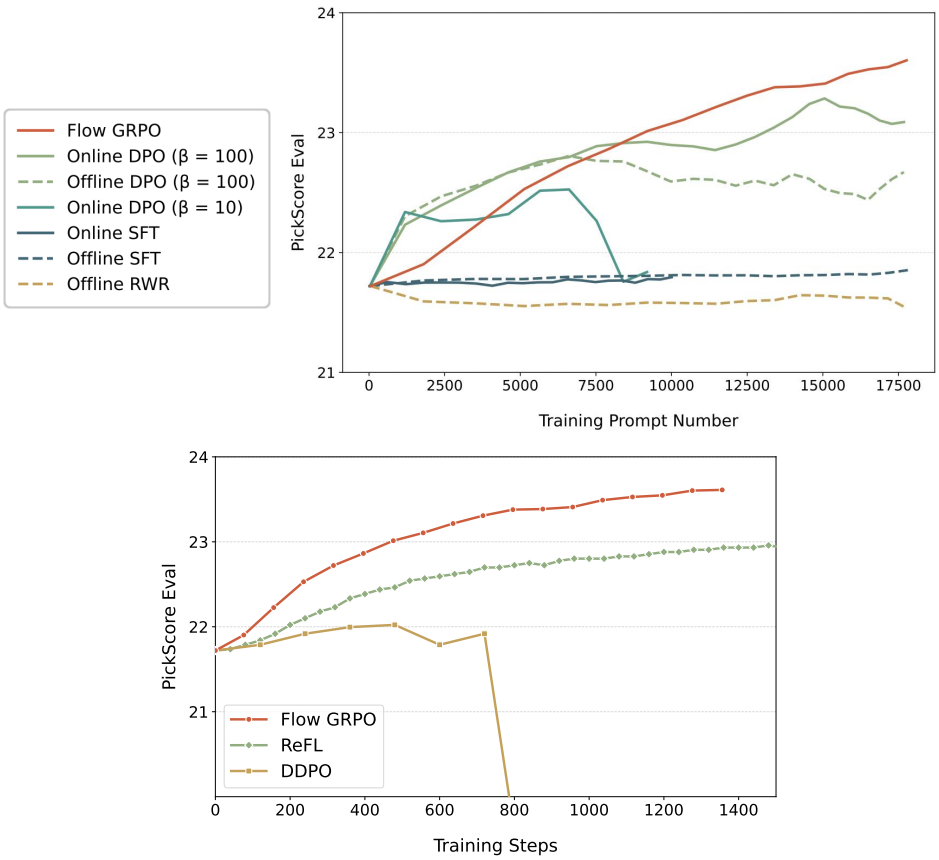


Figure 8: Comparison of Flow-GRPO and Other Alignment Methods on the Human Preference Alignment task. Since methods like DPO use different tuned batch sizes from Flow-GRPO, we use the number of training prompts on the x-axis for a fair comparison across these methods.

DDPO. DDPO [12] was originally developed for diffusion-based backbones, so we adapted it to flow-matching models via our ODE-to-SDE conversion. Using SD3.5-M as the base model and PickScore as the reward signal, we track the evaluation reward throughout the entire training process in Figure 8. We find that DDPO's reward increases more slowly than Flow-GRPO's and eventually collapses in the later stages, whereas Flow-GRPO trains stably and continues to improve consistently over time.

ReFL. ReFL [32] directly fine-tunes diffusion models by viewing reward model scores as human preference losses and back-propagating gradients to a randomly-picked late timestep t. Following ImageReward [32], we back-propagate gradients to a randomly chosen late timestep $t \in [30, 40]$ during denoising. Figure 8 shows that GRPO surpasses ReFL when the reward is differentiable, indicating that GRPO maintains strong performance in settings where ReFL applies. More importantly, GRPO does not require differentiable rewards, enabling direct use of state-of-the-art Vision-Language Models (VLMs) as reward providers. This offers two key advantages:

- **Sophisticated, General-Purpose Rewards:** VLMs can conduct human-like evaluations through a structured reasoning process. Given a prompt, a VLM can decompose it into key criteria, reason step by step to verify each aspect in the generated image, and then provide a comprehensive overall score. This enables a single, unified reward model to handle diverse tasks, from text-to-image generation to complex instruction-based image editing.
- **Future-Proof and Cost-Free Upgrades:** The field of VLMs is advancing at a breathtaking pace. By using a VLM as the reward source, our framework automatically benefits from these

improvements. As VLMs become more capable, the reward model becomes stronger without any additional training data or computational cost.

ORW. ORW [35] is an online reward-weighted regression method that guides the model to prioritize high-reward regions. Unlike KL regularization, it employs Wasserstein-2 regularization to prevent policy collapse and maintain diversity. To ensure a fair comparison, we adopt the same experimental setup as in our Human Preference Alignment task. For ORW, we set $\beta = 0.5$ and $\alpha = 1$ (lower values led to unstable training). The steps_per_epoch parameter, which controls how frequently the data-collecting policy is updated, was chosen from $20, 40, 100, 400$ based on best performance. Table 5 reports reward scores on the test set across training steps. Following ORW's Table 1, we randomly sampled 50 DrawBench prompts and generated 64 images per prompt to compute CLIP and Diversity scores. As shown in Table 6, Flow-GRPO outperforms ORW on both metrics.

Table 5: Reward scores on the test set over training steps.

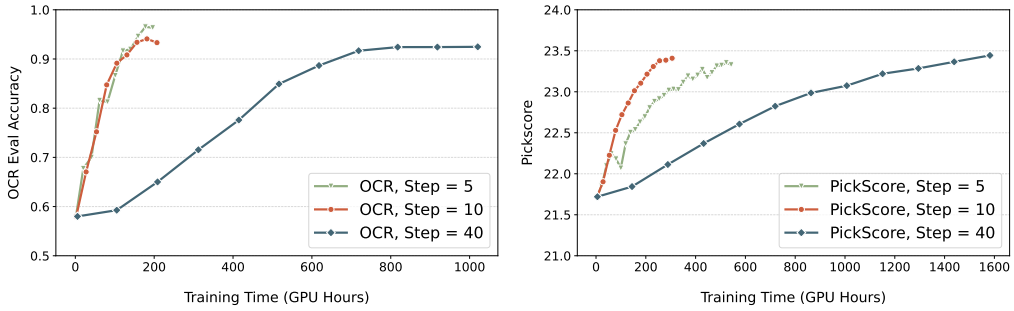
Method	Step 0	Step 240	Step 480	Step 720	Step 960
SD3.5-M + ORW	28.79	29.05	29.15	27.58	23.05
SD3.5-M + Flow-GRPO	28.79	**29.10**	**29.17**	**29.51**	**29.89**

Table 6: Comparison of CLIP and diversity scores across different fine-tuning methods.

Method	CLIP Score ↑	Diversity Score ↑
SD3.5-M	27.99	0.96
SD3.5-M + ORW	28.40	0.97
SD3.5-M + Flow-GRPO	**30.18**	**1.02**

C.2 Effect of Denoising Reduction

We show the extended Denoising Reduction ablations of Visual Text Rendering and Human Preference Alignment tasks in Figure 9.



(a) Visual Text Rendering

(b) Human Preference Alignment

Figure 9: Effect of Denoising Reduction

C.3 Effect of Initial Noise

We initialize each rollout with difference random noise to increase exploratory diversity during RL training. We perform an additioanl ablation to confirm this claim. With SD3.5-M as the base model and PickScore as the reward, we compare Flow-GRPO with different initial noise against Flow-GRPO with the same initial noise. Figure 10 shows the variant with different noise consistently achieved high rewards during the training process.

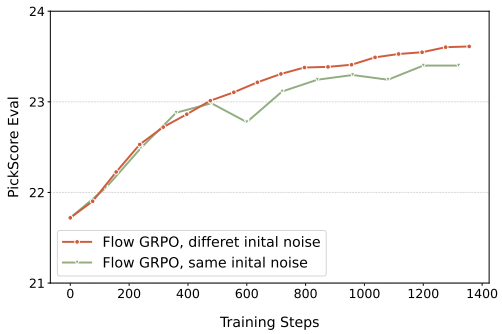


Figure 10: Effect of Initial Noise

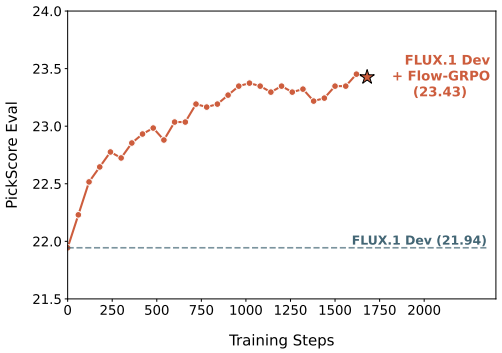


Figure 11: Additional Results on FLUX.1-Dev

C.4 Additional Results on FLUX.1-Dev

We run Flow-GRPO on FLUX.1-Dev [5] using PickScore as the reward signal. The reward curve rises steadily throughout training without noticeable reward hacking. Figure 11 shows the reward values over the training process, and Table 7 compares FLUX.1-Dev with FLUX.1-Dev + Flow-GRPO on DrawBench.

Table 7: Comparison of FLUX.1-Dev and Flow-GRPO fine-tuned models.

Model	Aesthetic	DeQA	ImageReward	PickScore	UnifiedReward
FLUX.1-Dev	5.71	4.31	0.85	22.62	3.65
FLUX.1-Dev + Flow-GRPO	6.02	4.24	1.32	23.97	3.81

C.5 Learning Curves with or without KL

Figure 12 shows learning curves for three tasks, with and without KL. These results emphasize that KL regularization is not empirically equivalent to early stopping. Adding appropriate KL can achieve the same high reward as the KL-free version and maintain image quality, though it requires longer training.

C.6 Additional Qualitative Results

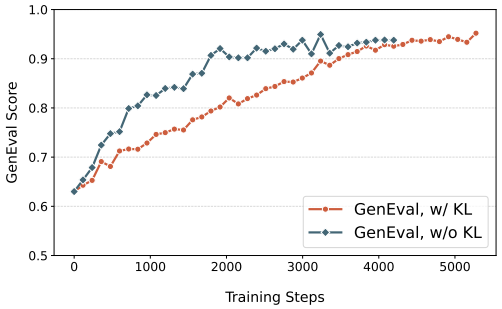
Figures 13, 14 & 15 qualitatively compare SD3.5-M with its Flow-GRPO enhanced versions (with and without KL regularization) using GenEval, OCR and PickScore rewards, respectively. Flow-GRPO with KL regularization improves the target capability while maintaining image quality and minimizing reward-hacking. Conversely, removing the KL constraint significantly degrades image quality and diversity.

C.7 Evolution of Evaluation Images During Flow-GRPO Training

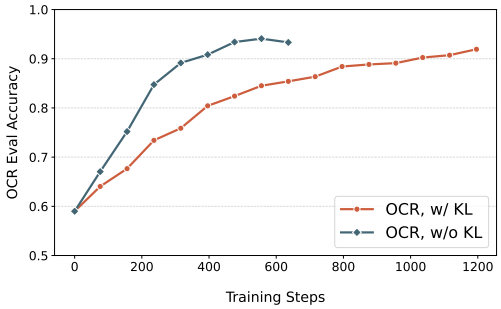
To better understand the training dynamics of our proposed Flow-GRPO framework, we visualize the evolution of generated samples corresponding to fixed evaluation prompts at regular intervals during training in Figure 16, 17 & 18. For consistency, all visualizations are produced using a 40-step ODE-based sampling schedule. These qualitative results provide a visual representation of how the model progressively improves its generation quality and alignment with task objectives over time.

D Training Sample Visualization with Denoising Reduction

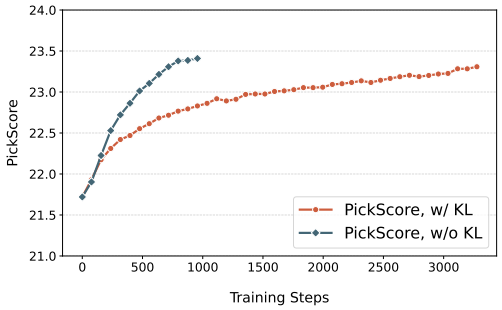
In this section, we compare images obtained with SDE sampling at various steps against those produced by ODE sampling, and offer an intuitive view of the denoising reduction strategy. Figure 19 presents SD3.5-Medium samples under four inference settings: (a) ODE sampling with 40 steps; (b) SDE sampling with 40 steps; (c) SDE sampling with 10 steps; (d) SDE sampling with 5 steps.



(a) Compositional Image Generation



(b) Visual Text Rendering



(c) Human Preference Alignment

Figure 12: Learning Curves with and without KL. KL penalty slows early training yet effectively suppresses reward hacking.

The 40-step ODE and SDE runs yield visually indistinguishable images, confirming that our SDE sampler preserves quality. Shortening the SDE schedule to 10 and 5 steps introduces conspicuous artifacts, like color drift and fine details blur. Contrary to expectation that such low-quality samples might hinder optimization. it actually do just the opposite and accelerate optimization. Because Flow-GRPO relies on relative preferences, it still extracts a useful reward signal, while the shorter trajectories signifactly cut wall-clock time. Consequently, Flow-GRPO with denoising reduction strategy converges more quickly on both layout-oriented benchmarks such as GenEval and quality-focused metrics such as PickScore, without sacrificing final performance.

Figure 13: Additional Qualitative comparison between the SD3.5-M and SD3.5-M + Flow-GRPO trained with **GenEval** reward.

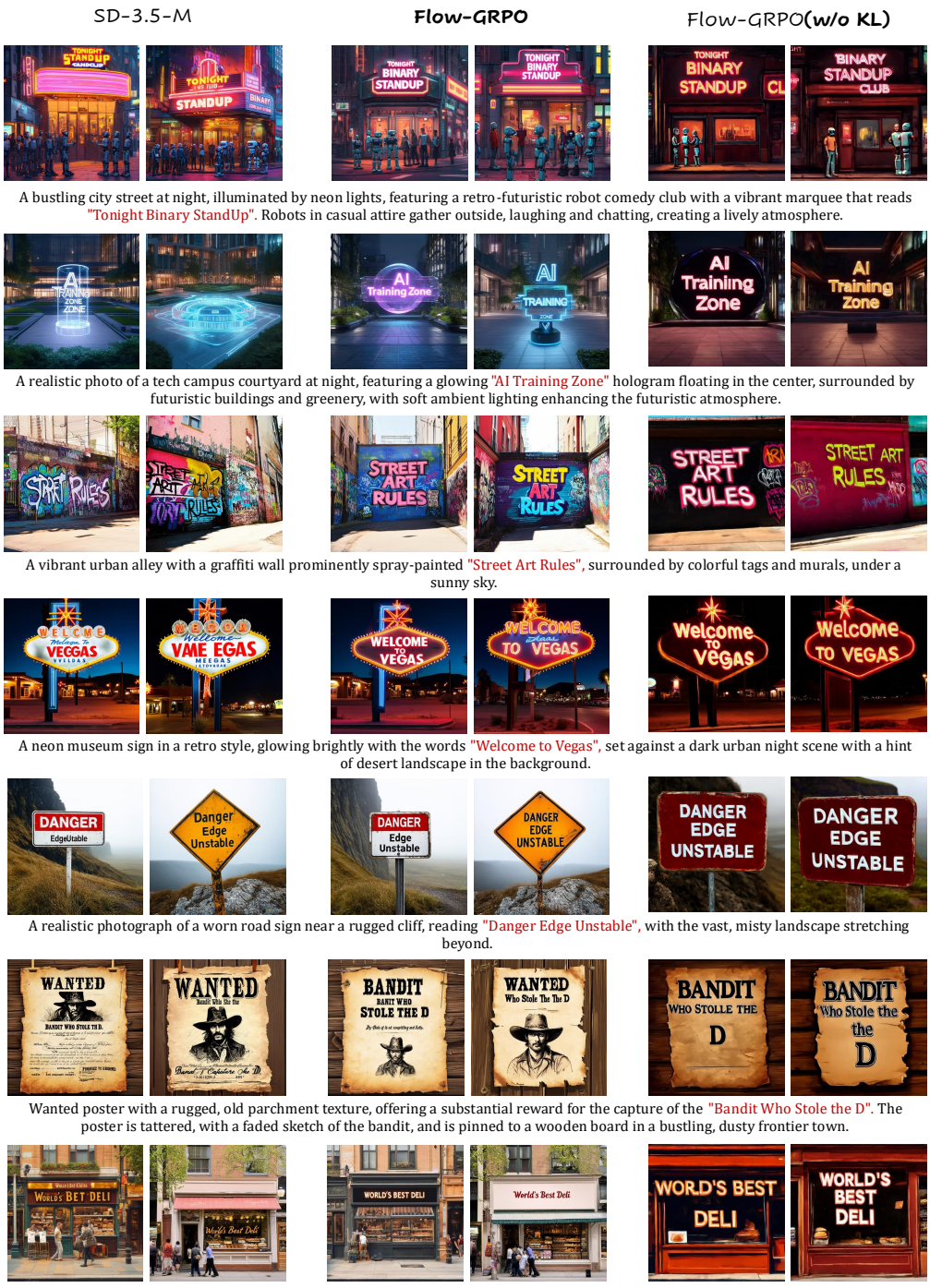


Figure 14: Additional Qualitative comparison between the SD3.5-M and SD3.5-M + Flow-GRPO trained with **OCR** reward.

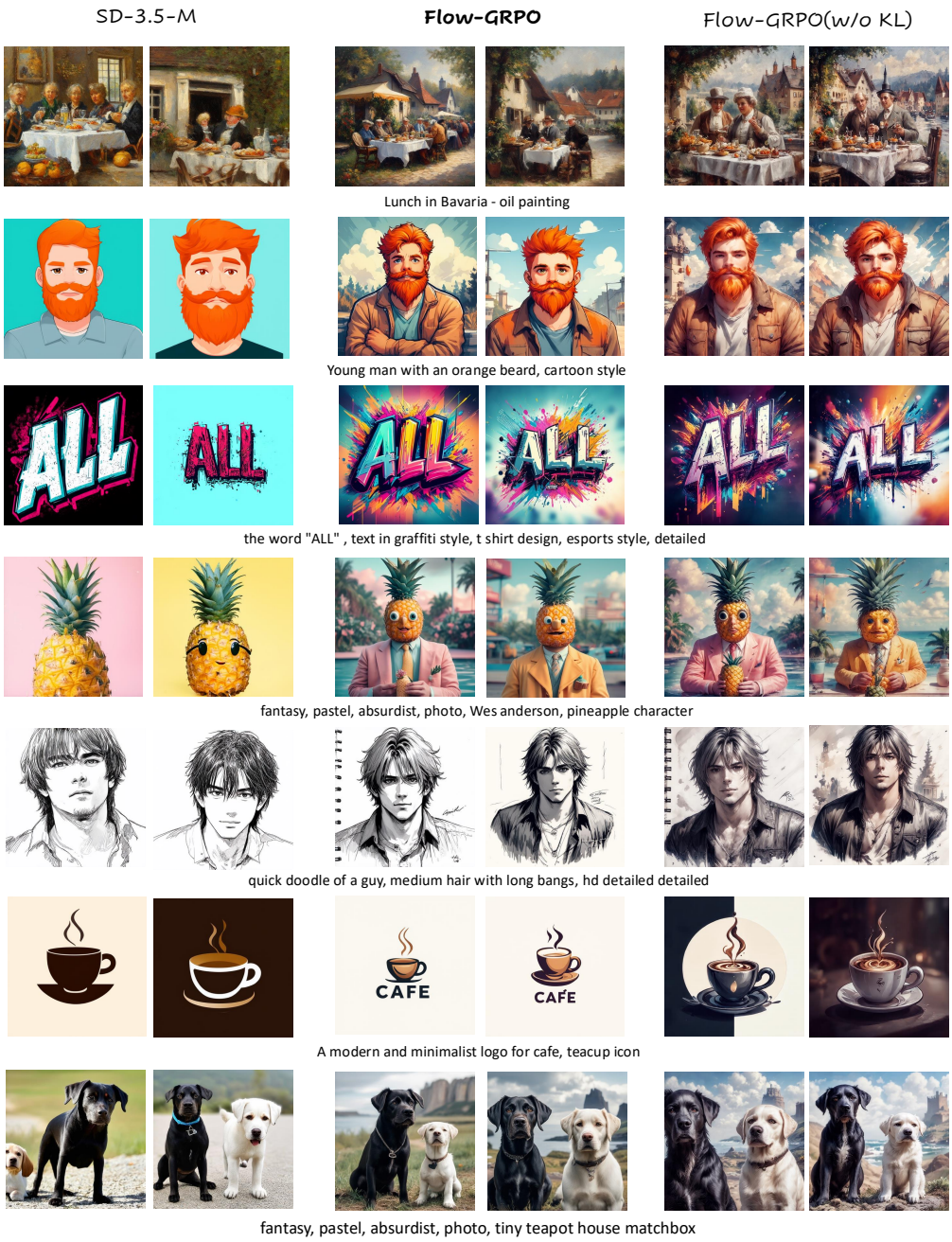


Figure 15: Additional Qualitative comparison between the SD3.5-M and SD3.5-M + Flow-GRPO trained with **PickScore** reward.

Figure 16: We visualize the generated samples across successive training iterations during the optimization of SD3.5-Medium on the **GenEval** task.

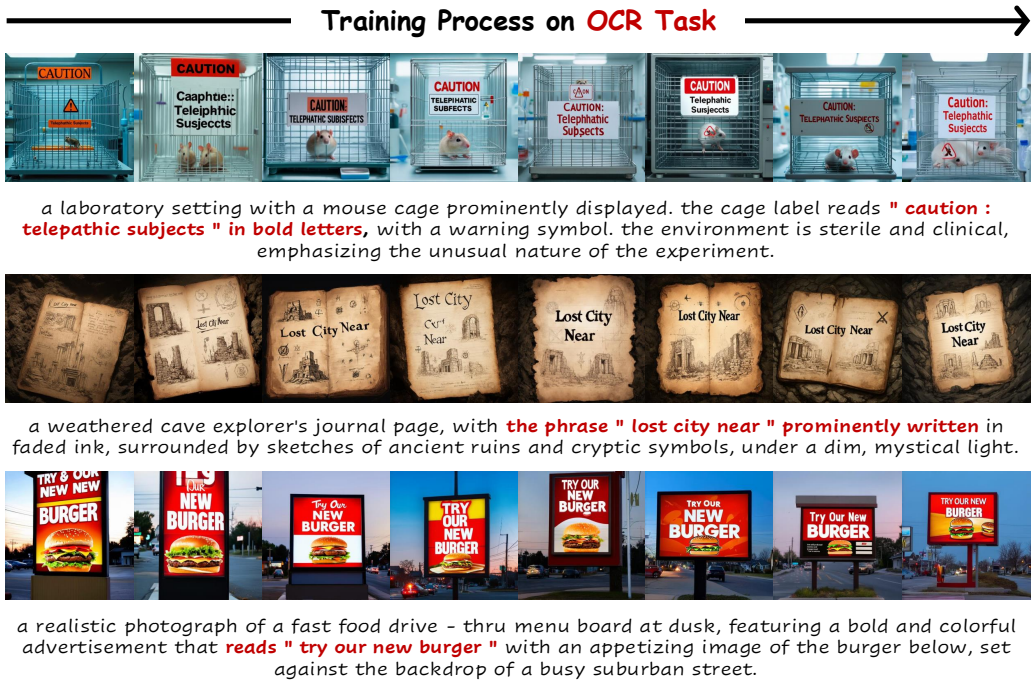


Figure 17: We visualize the generated samples across successive training iterations during the optimization of SD3.5-Medium on the **OCR** task.

Figure 18: We visualize the generated samples across successive training iterations during the optimization of SD3.5-Medium on the **PickScore** task.

Figure 19: Visualization of training samples under difference inference settings.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: Yes. Our main contribution are detailed in Sec. 1.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Yes, please see Sec. 6 for limitation.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: Yes. We provide the detailed derivations of SDE sampling in flow models, see Appendix A of our attached file.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We detail the experimental settings in Sec 5.1. We believe our experimental results can be easily reproduced.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: We provide code and data in the supplementary material, along with detailed instructions for installation and usage.

Guidelines:

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

6. **Experimental setting/details**

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

Answer: [Yes]

Justification: We detail the experimental settings in Sec 5.1.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [No]

Justification: Due to the resource limitation, we do not report error bars. As reported in Appendix B.4, we spent numerous resources for our experiments, which makes it prohibitively expensive to run each experiments for multiple times.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We report the GPU resources used in our experiments in Appendix B.4, and provide GPU hour statistics in Fig. 9.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We followed the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [No]

    Justification: This work focuses on publicly available academic benchmarks and datasets (e.g., GenEval), and does not involve any private or personal data. So we do not identify any explicit negative societal impacts arising from this work.

    Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [No]

    Justification: We do not foresee any high risk for misuse of this work.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Yes, we credited them in appropriate ways.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [NA]

    Justification: This paper does not release new assets.

    Guidelines:

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

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

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

    Answer: [NA]

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

    Guidelines:

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

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

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

    Answer: [NA]

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

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [NA]

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

    Guidelines:

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