DFS-GRPO: REWARD GUIDED TREE SEARCH LEADS TO PROVABLE IMPROVEMENT IN DIFFUSION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose DFS-GRPO, a reward-guided tree search method that introduces reward-guided multi-tree depth-first search (DFS) sampling to improve diffusion post-training by directing a fixed rollout budget toward high-reward regions while preserving exploration. DFS sampling uses rewards from completed rollouts to select and reuse promising denoising prefixes, progressively directing the sampling groups toward high-reward regions. We combine this sampler with initial noise truncation to strengthen exploitation, while subtree budget allocation and retention of multiple promising branches preserve exploration. An idealized local analysis shows that reward-based selection biases the expected state update toward the gradient of expected reward. Experiments on SD3-Medium and FLUX.1-dev demonstrate that DFS-GRPO improves reward optimization and maintains exploration to guarantee training stability. With matched training rewards, DFS-GRPO outperforms the strongest Flow-GRPO-S3 baseline (ODE–SDE sampling, tree sampling, and noise truncation) across three benchmarks on FLUX.1-dev, achieving relative improvements of 2.4% on UniGenBench, 5.1% on GenEval, and 5.6% on LongText-Bench (English) over this baseline. Under standard inference, the resulting model reaches 0.8407 on OneIG-Bench Alignment, comparable to the newer-generation Z-Image-Turbo and to Imagen-3.

1 INTRODUCTION

Rectified flow (RF)-based diffusion models have achieved strong performance in image, video, and 3D generation with large-scale pre-training data (Liu et al., 2022; Esser et al., 2024; Li et al., 2024; Team et al., 2025). To better align these models with human preferences, current works adopt post-training algorithms such as DPO (Wallace et al., 2024) and PPO (Black et al., 2023).

Recently, GRPO-based algorithms (Liu et al., 2025a; Xue et al., 2025) have significantly improved the performance of base models. With stochastic differential equation (SDE) sampling, RF models generate diverse samples from the same prompt, and their relative rewards guide policy optimization. A key part of this RL process is balancing exploration and exploitation (E-E balance) (Sutton et al., 1998). Recent works adopt mixed ODE–SDE sampling (Li et al., 2025a; He et al., 2025; Zhou et al., 2025) or tree structures (Li et al., 2025b; Fu et al., 2025) for efficient exploration and faster GRPO training. However, repeated runs of Flow-GRPO-S3 with tree sampling still show large reward drops (Figure 1). We view insufficient exploitation of high-reward samples as one factor behind this instability, reflecting an imperfect E-E balance, where rollout rewards guide policy updates but not the expansion of promising denoising prefixes.

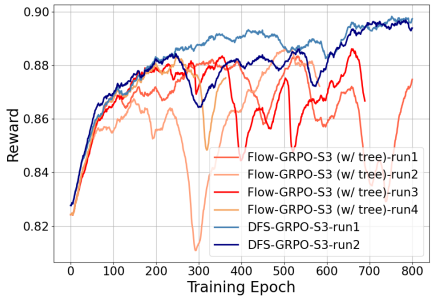


Figure 1: **Stable reward improvement with DFS.** Across repeated runs, DFS-GRPO-S3 sustains higher rewards with smaller drops than Flow-GRPO-S3 (w/ tree).

To improve the E-E balance and fully exploit high-reward samples, we direct a fixed rollout budget toward high-reward regions while maintaining exploration. Sampling randomness comes from two sources, SDE noise and initial noise, and we address both. Regarding SDE noise, actively searching over noise choices can improve sample quality (Ma et al., 2025). Since diffusion models denoise

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



Figure 2: Qualitative Results on LongText-Bench for DFS-GRPO on FLUX.1-Dev with Multi-Reward Training (HPSv2.1 and Clipscore).

sequentially, noise at an earlier SDE step shapes all later states, so a promising prefix is worth reusing. Hence, we propose **DFS-GRPO**, whose core is reward-guided multi-tree depth-first search (DFS) sampling (Figure 3, red region). More specifically, DFS scores denoising prefixes using completed-rollout rewards; each new sub-tree reuses promising prefixes and samples fresh noise at later SDE steps, progressively directing sampling toward high-reward regions. For initial noise, inspired by the truncation trick in BigGAN (Brock et al., 2018), we trade diversity for fidelity by truncating the initial noise distribution during training, which further strengthens exploitation.

Concentrating sampling also narrows the range of trajectories explored. To maintain exploration, DFS-GRPO allocates the rollout budget across sub-trees, balancing broad initial search with later expansion of promising prefixes. It retains the top two prefixes so alternative continuations remain available (Figure 3, blue region; Figure 5b). Together, these choices balance high-reward exploitation with continued exploration, yielding more stable training and higher rewards (Figure 1).

Theoretically, we analyze DFS-GRPO from three perspectives. From the search direction perspective, our local analysis shows that DFS selection provides implicit guidance along the expected-reward gradient. From the reward confidence perspective, our Gaussian-mixture analysis explains how this guidance increases confidence in high-reward regions. From the policy optimization perspective, we analyze the convergence of idealized policy updates and explain how reward-guided rollouts provide learning signals (Sec. 4.1).

Empirically, DFS-GRPO improves all four in-domain rewards over Flow-GRPO-S3 (w/ tree) on both SD3-Medium and FLUX.1-dev (Table 1). For out-of-domain performance on FLUX.1-dev, DFS-GRPO trained with HPS-v2.1 and CLIPScore surpasses the strongest baseline, Flow-GRPO-S3 with ODE–SDE sampling, tree structure, and truncation, on UniGenBench, GenEval, and LongText-Bench (English). Adding OCR and spatial rewards with structured prompts raises the relative improvements over this baseline to 2.4%**,** 5.1%**, and** 5.6%, respectively (Tables 2 and 3). Under standard single-LoRA inference, the resulting FLUX.1-dev model reaches 0.8407 on OneIG-Bench Alignment, comparable to Z-Image-Turbo and Imagen 3 (Table 4). Furthermore, we extend DFS-GRPO to video generation, where it also improves over Flow-GRPO-S3. To summarize:

- We propose DFS-GRPO, centered on reward-guided multi-tree DFS sampling that reuses promising denoising prefixes to search high-reward regions. We balance exploitation and exploration: DFS sampling and initial noise truncation strengthen exploitation, while sub-tree budget allocation and Top-2 branch retention preserve exploration.
- We provide a theoretical analysis from the perspectives of search direction, reward confidence, and policy optimization to show why DFS-GRPO achieves great performance.
- We show that DFS-GRPO outperforms Flow-GRPO-S3 baselines in reward optimization and benchmark performance, brings FLUX.1-dev into the performance range of newer image models, and extends to video generation.

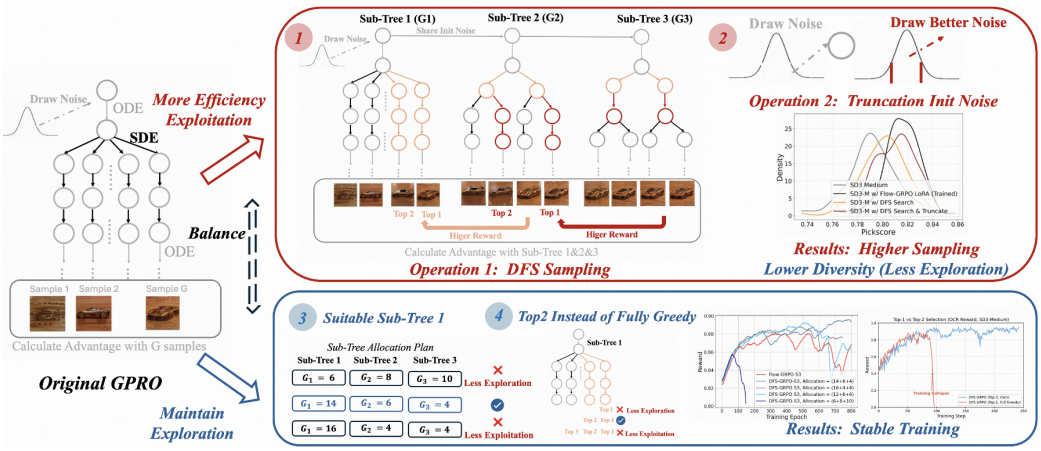


Figure 3: **Balancing exploitation and exploration in DFS-GRPO.** Black and gray arrows denote SDE and ODE steps, respectively. Exploitation (red): (1) DFS sampling reuses promising denoising prefixes across sub-trees, and (2) initial noise truncation concentrates sampling in high-reward regions, shifting SD3-Medium rewards upward before training. Exploration (blue): (3) sub-tree budget allocation balances broad initial sampling with expansion of promising prefixes, and (4) Top-2 branch retention keeps alternative continuations. Allocation curves show stable reward optimization; on OCR, Top-2 sustains learning while fully greedy Top-1 training collapses.

2 Related Works

Sampling and Exploration for Diffusion RL. Flow-GRPO (Liu et al., 2025a) and DanceGRPO (Xue et al., 2025) introduce GRPO-based post-training for diffusion and flow models. Subsequent works improve training efficiency through sampling design. MixGRPO (Li et al., 2025a) and TempFlow-GRPO (He et al., 2025) mix ODE and SDE steps to focus stochastic exploration, while Branch-GRPO (Li et al., 2025b) and Dynamic-TreeRPO (Fu et al., 2025) organize rollouts into trees with shared prefixes. Regarding reward-guided sampling, Smart-GRPO (Yu et al., 2025) iteratively refines a noise distribution using rewards from one-step image previews, and Pro-GRPO (Ge et al., 2026) expands and prunes trajectory groups to retain reward variation. To preserve generation diversity, DiverseGRPO (Liu et al., 2026) combines diversity-aware rewards with early-stage regularization. Our DFS-GRPO ranks denoising prefixes using completed-rollout rewards, then reuses promising prefixes and branches at later SDE steps through multi-tree depth-first search. Initial noise truncation strengthens exploitation; sub-tree budget allocation sustains early search breadth, and Top-2 retention preserves alternative continuations.

Reward Feedback and Policy Optimization. G^2RPO (Zhou et al., 2025) integrates advantages from multi-granularity ODE continuations to evaluate individual SDE steps, while Pref-GRPO (Wang et al., 2025b) uses win rates to construct group-relative advantages. DiffusionNFT (Zheng et al., 2025) and DGPO (Luo et al., 2025) construct forward-noising training objectives from reward-labeled samples, and GRPO-Guard (Wang et al., 2025a) introduces RatioNorm and Gradient Reweight to improve update stability. Recent work further develops step-wise rewards and noise calibration in DenseGRPO (Deng et al., 2026), process-reward-gradient guidance for video in Euphonium (Zhong et al., 2026), and efficient single-step video policy optimization in Flash-GRPO (He et al., 2026). We retain the Flow-GRPO objective and improve the rollout groups supplied to it through reward-guided prefix search.

3 Preliminary

RF and MDP Modeling. Since the GRPO algorithm involves the SDE, we introduce RF from the general form (Song et al., 2020). Let the target distribution be p^*. Then, the forward process $\{p_t\}_{t\in[0,T]}$ has the following form:

$$\mathrm{d}x_t^{\rightarrow} = -f(t)x_t^{\rightarrow}\,\mathrm{d}t + g(t)\mathrm{d}B_t, \quad x_0^{\rightarrow} \sim p^*(\cdot) \in \mathbb{R}^d\,,$$

where $f(t)$ and $g(t)$ is non-negative non-decreasing sequence and $(B_t)_{t\geq 0}$ is a d-dimensional Brownian motion. As shown in Singh & Fischer (2024), with $f(t) = 1/(1-t)$ and $g(t) = \sqrt{2t/(1-t)}$, we have $p_t(x_t^{\rightarrow}|x_0^{\rightarrow}) = \mathcal{N}(x_t^{\rightarrow}; (1-t)x_0^{\rightarrow}, t^2 I_d)$, which is called rectified flow.



Figure 4: Qualitative Comparison with Single Reward. With HPS-v2.1 in the training phase, we show that our DFS-GRPO (Truncate) with FLUX.1-dev base models and strong baseline Flow-GRPO-S3 (w/ tree) in terms of text-image alignment, aesthetic, and text rendering.

To generate samples, diffusion models reverse the forward process and obtain the reverse process:

$$\mathrm{d}x_t = \left[f(T-t)x_t + \frac{1+\alpha^2}{2} g(T-t)^2 \nabla \log p_{T-t}(x_t)\right]\mathrm{d}t + \alpha g(T-t)\,\mathrm{d}B_t\,, \tag{1}$$

where $(x_t)_{t\in[0,T]} = \left(x_{T-t}^{\rightarrow}\right)_{t\in[0,T]}$ is the variable of the reverse process and $\alpha \in [0,1]$. For parameter α, it is used to determine the type of the reverse process. When $\alpha = 0$, it is a deterministic process, called the probability flow ODE (PFODE). Otherwise, this process is a stochastic process (reverse SDE). To sample with the above continuous reverse process, we usually discretize the continuous reverse process into K intervals $\{t_k\}_{k=0}^{K}$ and use sampling algorithms (such as the Euler method) with the learned score function $s_\theta(\cdot, t)$. In the following part, we abbreviate x_{t_k} to x_k.

As shown in Liu et al. (2025a), the discretized reverse process can be viewed as a MDP with state $\boldsymbol{s}_k \triangleq (c, t_k, x_k)$, where c is the given prompt, action $\boldsymbol{a}_k \triangleq x_{k+1}$ and the policy $\pi_\theta(\boldsymbol{a}_k|\boldsymbol{s}_k) \triangleq p_\theta(x_{k+1}|x_k, c)$.

Flow-GRPO. Let R be a reward model. Inspired by PPO, Flow-GRPO maximizes the following policy gradient objective function to maximize reward:

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

where $f(r, \hat{A}, \theta, \epsilon, \beta)$ has the following form

$$\frac{1}{G}\sum_{i=1}^{G}\frac{1}{K}\sum_{k=0}^{K-1}\left(\min(r_k^i(\theta)\hat{A}_k^i, \text{clip}\left(r_k^i(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_k^i) - \beta D_{\text{KL}}\left(\pi_\theta \| \pi_{\text{ref}}\right)\right),$$

with $r_k^i(\theta) = \frac{p_\theta\left(x_{k+1}^i|x_k^i,c\right)}{p_{\theta_{\text{old}}}\left(x_{k+1}^i|x_k^i,c\right)}$, K is the denoised step number, and $\hat{A}_k^i$ is an advantage function for the i-th samples determined by the samples group $\{x_K^i\}_{i=1}^{G}$ given a prompt c (recall that x_{t_K} is the clean samples generated by the discretized reverse process):

$$\hat{A}_k^i = \frac{R\left(x_0^i, c\right) - \text{mean}(\left\{R(x_0^i, c)\right\}_{i=1}^{G})}{\text{std}\left(\left\{R\left(x_0^i, c\right)\right\}_{i=1}^{G}\right)}, \forall k \in \{0, ..., K\}\,.$$

For the sampling phase of Flow-GRPO, the algorithm adopts reverse SDE (Eq. 1, $\alpha \neq 0$) for $\forall k \in [0, K-1]$ to generate diverse samples given the same prompt. Recently, a series of works show that only use SDE sampling step for $k \in [k_{\text{SDE}}, k_{\text{SDE}} + K_s]$ and adopt PFODE step otherwise, where k_{SDE} the SDE starting step and K_s is the number of SDE steps. Our DFS-GRPO also adopts this operation.

4 DFS-GRPO

DFS-GRPO directs a fixed rollout budget toward high-reward regions while maintaining exploration for continued learning. Its core is reward-guided multi-tree depth-first search (DFS) sampling, which reuses promising denoising prefixes to strengthen exploitation; initial noise truncation complements this search at initialization. These two exploitation operations trade some trajectory diversity for higher generation quality. To maintain exploration, we introduce sub-tree budget allocation and Top-2 branch retention (Figure 3). We retain the Flow-GRPO objective and redesign the rollout groups supplied to it.

Exploitation through multi-tree DFS sampling. Each rollout follows an ODE–SDE–ODE schedule; DFS searches over its stochastic steps from higher to lower noise levels (Figure 3, operation 1). For a fixed prompt and initial noise, the first sub-tree generates complete trajectories with independent noise at each SDE step. The terminal reward of each rollout scores its prefix through the first SDE step, and we retain the two highest-scoring prefixes. The second sub-tree reuses these prefixes, draws fresh noise from the next SDE step onward, and completes the new trajectories to obtain their rewards. Later sub-trees repeat this process with progressively longer retained prefixes. Each round thus reuses stochastic decisions that led to high rewards while exploring how to continue them, directing the remaining budget toward promising denoising regions.

All completed rollouts across the sub-trees, including lower-reward outcomes from the first sub-tree, form the group used to compute relative advantages. This preserves the reward contrasts used to reinforce better continuations. DFS search improves sampling rewards even before training (Table 1); during training, it supplies high-reward trajectories for policy updates, coupling reward-guided search with policy improvement.

Initial noise truncation. As a complement to DFS sampling, we concentrate the starting noise using the truncation strategy of Brock et al. (2018):

$$x_0 \sim \mathcal{TN}(0, I_d; -a, a), \qquad a = 1.5. \tag{2}$$

Here $\mathcal{TN}$ denotes a standard Gaussian conditioned on the hypercube $[-a, a]^d$. This further shifts the observed reward distribution toward higher values (Figure 3, operation 2). Truncation is used during training; evaluation uses standard Gaussian initialization. The coordinate-wise density is given in Appendix B.2.

Exploration through sub-tree allocation and Top-2 retention. Since DFS sampling and initial noise truncation trade some diversity for generation quality, we use sub-tree budget allocation to preserve broad initial exploration and Top-2 retention to keep alternative continuations available.

Sub-tree budget allocation balances broad initial sampling with focused prefix expansion (Figure 3, operation 3). The first sub-tree explores before any prefix is fixed; too few samples restrict the candidate paths, while too many leave little budget for later expansion. With $G = 24$ rollouts and three SDE steps, we use $(G_1, G_2, G_3) = (14, 6, 4)$, where G_j is the total number of rollouts in sub-tree j. The allocation ablation in Figure 6(b) examines this balance.

Top-2 branch retention carries forward two promising prefixes rather than only the highest-reward one, preserving more diverse continuations than fully greedy Top-1 selection as the search deepens (Figure 3, operation 4). The OCR ablation shows sustained learning with Top-2 and training collapse under fully greedy Top-1 selection (Figure 5b). Algorithms 1 and 2 give the full procedure.

4.1 Understanding Reward-Guided Search

DFS-GRPO concentrates sampling on promising denoising prefixes while preserving alternative continuations for learning. We explain this design through three connected views: the direction induced by reward-based selection, its interpretation in a reward-based Gaussian-mixture model, and the role of the resulting rollouts in policy optimization.

Reward-based selection gives a search direction. Fix a prompt, a denoising stage, and the current policy. Let Y denote the completed sample and $R(Y)$ its terminal reward. The continuation value $J(x) = \mathbb{E}[R(Y) \mid x]$ measures the expected reward obtained by continuing from state x. DFS uses rewards from completed rollouts to score candidate prefixes; to isolate the effect of selection, we analyze ranking by their exact continuation values. Consider n candidate states generated by a local

stochastic denoising step:

$$x_i^+(h) = x + bh + \nu\sqrt{h}\,\varepsilon_i, \qquad \varepsilon_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_d). \tag{3}$$

Here b is the local denoising drift, $\nu > 0$ is the noise amplitude, and h is the step size; candidates share the drift and differ in their noise. Select $i^* \in \arg\max_{1\le i\le n} J(x_i^+(h))$, breaking ties by index, and write $x_{\text{sel}}^+ = x_{i^*}^+(h)$.

Theorem 4.1 (Local directional alignment)**.** *Fix a finite $n \ge 2$. If J is differentiable at x and $\nabla J(x) \neq 0$, then, as $h \downarrow 0$,*

$$\mathbb{E}[x_{\text{sel}}^+ - x \mid x] = bh + \nu\sqrt{h}\,\lambda_n \frac{\nabla J(x)}{\|\nabla J(x)\|} + o(\sqrt{h}), \tag{4}$$

where $\lambda_n = \mathbb{E}[\max_{i\le n} Z_i] > 0$ for independent $Z_i \sim \mathcal{N}(0,1)$.

A randomly chosen candidate has mean displacement bh. Reward-based selection adds a component along $\nabla J(x)$: noise pointing toward better continuations is preferentially retained, while orthogonal components average to zero to leading order. This explains how comparing rollout values supplies a search direction without differentiating the reward model, consistent with the sampling gains in the red region of Figure 3 and in Table 1.

GMM connects this direction to high-reward regions. We model the reward-induced sample distribution p^r, whose modes represent regions of different reward levels, rather than the pretraining data distribution. At a fixed noise level, the two-mode model is $p^r(x) = w_-\mathcal{N}(x;\mu_-,\Sigma) + w_+\mathcal{N}(x;\mu_+,\Sigma)$. Here μ_- and μ_+ are the low- and high-reward component means, $\Sigma \succ 0$ is their shared covariance, and $w_-, w_+ > 0$ are mixture weights summing to one. Gaussian forward noising preserves this mixture form; the means and covariance are understood at the fixed noise level. The reward confidence $P_+(x)$ is the posterior probability that state x belongs to the high-reward component. In this model, assign each completed sample its high-reward posterior as terminal reward. Sampling the completed sample from its model conditional distribution given x then gives $J(x) = P_+(x)$, and $\nabla J(x) = P_+(x)(1 - P_+(x))\Sigma^{-1}(\mu_+ - \mu_-)$. Thus the selection direction coincides with the reward-guidance direction $\nabla \log P_+(x)$: both align with the covariance-scaled direction from the low-reward mean to the high-reward mean.

To quantify the corresponding confidence change, compare two states x and x' at the same noise level. Define $\Delta_{\text{odds}} = (\mu_+ - \mu_-)^\top\Sigma^{-1}(x' - x)$, the increase in the high-versus-low log posterior odds. Since these log odds are affine in the state, Theorem 4.1 shows that selection increases their expected value relative to a randomly chosen candidate, to leading order. When $\Delta_{\text{odds}} \ge 0$,

$$P_+(x') \ge \frac{P_+(x)}{P_+(x) + (1 - P_+(x))e^{-\Delta_{\text{odds}}}} \ge P_+(x). \tag{5}$$

The first relation is an equality for two modes. For multiple modes, the same lower bound holds when Δ_{odds} is the smallest high-reward-versus-competitor log-odds increase and is nonnegative. The bound quantifies how a given log-odds gain translates into increased reward confidence. The local-selection proof, full GMM scores and posteriors, and guided-process analysis are developed in Appendix C.

From selected rollouts to policy optimization. Returning to the training reward R, $J(x)$ describes continuation quality with the current policy held fixed. The policy objective $V(\theta) = \mathbb{E}_{\pi_\theta}[R(Y)]$ instead measures expected reward as the denoising policy parameters θ change, averaging over initial noise and the subsequent trajectory. For an L-smooth V bounded above by V^*, exact gradient ascent with step size $1/L$ gives the standard bound after S updates (Agarwal et al., 2019):

$$\min_{0\le s<S} \|\nabla_\theta V(\theta_s)\|^2 \le \frac{2L\,[V^* - V(\theta_0)]}{S}. \tag{6}$$

Here L controls how rapidly the parameter gradient can change, and $V^* - V(\theta_0)$ is the initial reward gap. The bound describes progress toward a stationary point of the policy objective. DFS-GRPO acts on the rollout groups used to construct the updates: prefix reuse concentrates samples in high-reward regions, while subtree allocation and multi-branch retention maintain exploration (Figures 5b and 6). Group-relative advantages center rewards within each group; retaining lower-reward rollouts alongside promising ones supplies the reward contrasts used to reinforce better continuations. This connects directed sampling to the learning signal; training curves and GPU-hour comparisons characterize its optimization benefit (Figures 5a and 11). The smooth-ascent derivation is given in Appendix C.6.

Table 1: **Reward optimization on SD3-Medium and FLUX.1-dev.** DFS-GRPO improves all four rewards over Flow-GRPO-S3 (w/ tree). Best and second-best scores are bold and underlined within each backbone. Time is seconds per training iteration; * marks inference time per prompt.

Method	Time (s) ↓	PickScore ↑	HPS-v2.1 ↑	Aesthetic ↑	CLIPScore ↑
SD3-Medium (steps = 40)	/	21.84	0.300	5.96	0.290
+ Flow-GRPO (w/ tree)	65.30	23.37	0.357	7.57	0.298
+ Flow-GRPO-S3	52.16	23.09	0.360	7.52	0.294
+ Flow-GRPO-S3 (w/ tree)	44.09	23.19	0.370	8.46	0.300
+ DFS search (no training)	40.73*	22.64	0.327	6.36	<u>0.304</u>
+ DFS-GRPO-S3	51.31	<u>23.45</u>	<u>0.373</u>	<u>9.11</u>	**0.305**
+ DFS-GRPO-S3 (trunc.)	52.07	**23.47**	**0.374**	**9.35**	0.303
FLUX.1-dev (steps = 28)	/	22.15	0.305	6.23	0.278
+ Flow-GRPO (w/ tree)	82.26	22.80	0.361	8.57	0.287
+ Flow-GRPO-S3 (w/ tree)	64.11	23.10	0.365	9.46	0.293
+ DFS-GRPO-S3	69.29	<u>23.21</u>	<u>0.368</u>	<u>9.56</u>	<u>0.295</u>
+ DFS-GRPO-S3 (trunc.)	69.79	**23.31**	**0.371**	**9.58**	**0.298**

5 EXPERIMENTS

DFS-GRPO improves reward optimization on SD3-Medium and FLUX.1-dev, strengthens FLUX.1-dev across four benchmarks, and remains competitive with forward-process methods at comparable GPU budgets. Allocation and branch-retention ablations examine the two exploration mechanisms (Figure 3, blue region).

Models and baselines. We train SD3-Medium and FLUX.1-dev with a group size of 24. Our main sampler uses three SDE steps within an ODE–SDE–ODE trajectory, subtree allocation $(14, 6, 4)$, Top-2 prefix retention, and BoN 4 for selecting high- and low-reward samples for optimization. We build on Flow-GRPO (Liu et al., 2025a) with ODE–SDE sampling (Li et al., 2025a; He et al., 2025), shared-prefix tree sampling (Li et al., 2025b; He et al., 2025), and BoN selection (Xue et al., 2025). Our strongest benchmark baseline additionally uses initial noise truncation: **Flow-GRPO-S3 (tree + truncation)**. Post-trained models are evaluated from standard Gaussian noise with the standard sampler; truncation is a training operation. We additionally report inference-time DFS search without training in Table 1.

Rewards and benchmarks. We first optimize PickScore, HPS-v2.1, Aesthetic, and CLIPScore separately. For benchmark evaluation, we train FLUX.1-dev with HPS-v2.1 + CLIPScore on the Flow-GRPO prompt set, then extend this recipe with OCR and GroundingDINO spatial rewards and structured prompts. We evaluate UniGenBench (Wang et al., 2025b), GenEval (Ghosh et al., 2023), LongText-Bench (Geng et al., 2025), and the Alignment dimension of OneIG-Bench (Chang et al., 2025). Our post-trained FLUX.1-dev models use standard single-LoRA inference with 30 sampling steps; in-domain evaluation uses 40 steps for SD3-Medium and 28 for FLUX.1-dev. Additional settings and score breakdowns appear in the appendix.

5.1 STRONGER REWARD OPTIMIZATION AND BENCHMARK PERFORMANCE

Reward-guided sampling improves both backbones. DFS-GRPO improves all four in-domain rewards over Flow-GRPO-S3 with tree sampling on SD3-Medium and FLUX.1-dev (Table 1). On SD3-Medium, DFS-GRPO with truncation raises Aesthetic from 8.46 to 9.35; on FLUX.1-dev, it raises HPS-v2.1 from 0.365 to 0.371 and CLIPScore from 0.293 to 0.298. The training trajectories in Figures 9 and 10 show improved reward optimization across both backbones. This benefit also appears at comparable GPU time (Figure 11). DFS search alone improves the untrained SD3-Medium model across all four rewards. The reward distributions in the red region of Figure 3 show how DFS sampling and initial noise truncation shift sampling toward higher rewards before training.

Stronger FLUX.1-dev performance across benchmarks. With HPS-v2.1 + CLIPScore training, DFS-GRPO reaches 71.76 on UniGenBench, 73.00 on GenEval, and 0.696 on LongText-English, exceeding the strongest Flow-GRPO baseline on all three (Tables 2 and 3). Adding structured OCR and spatial rewards extends these gains to 72.81**,** 74.98**, and** 0.7184, respectively: relative

Table 2: **Out-of-Domain Comparison on UniGenBench and GenEval.** (†) Extended multi-reward training (HPS-v2.1 + CLIPScore + OCR + GroundingDINO) with structured prompts. All DFS-GRPO/Flow-GRPO variants use standard inference. (*) Results from Wang et al. (2025b). (‡) Independent models from the UniGenBench leaderboard. Best and second-best results within the FLUX.1-dev group are **bolded** and underlined.

	UniGenBench											GenEval
Method	**Overall**	Style	World.	Attr.	Act.	Relat.	Logic	Gram.	Comp.	Layout	Text	**Overall**
FLUX.1-dev based (Aug 2024) — ours and baselines												
FLUX.1-dev (Step=30)(*)	61.30	83.90	88.92	67.84	62.17	67.26	30.91	60.96	47.04	71.83	32.18	62.92
+ Flow-GRPO (*)	61.81	84.92	88.98	68.44	62.54	68.10	31.01	59.36	50.60	71.07	33.07	64.85
+ Flow-GRPO-S3 (w/ tree)	69.48	87.90	90.51	75.33	66.27	73.35	39.22	57.62	60.23	76.31	39.66	71.12
+ Flow-GRPO-S3 (w/ tree&Trun.)	71.11	**89.80**	92.09	75.00	69.66	75.76	39.45	59.36	62.31	75.37	40.52	71.34
+ Pref-GRPO (*)	69.46	88.40	90.35	75.00	69.77	**76.52**	**44.09**	**63.27**	62.42	**77.61**	47.13	70.53
+ DFS-GRPO-S3 (Truncate)	71.76	89.10	**93.83**	78.95	**70.04**	74.62	41.28	58.42	62.95	74.07	43.97	73.00
+ DFS-GRPO-S3 (Multi-Reward)†	**72.81**	88.80	92.72	**80.02**	68.89	75.89	39.91	61.90	**65.54**	75.75	**52.30**	**74.98**
Independent models from UniGenBench leaderboard‡ (2025–2026, for reference)												
LongCat-Image (2025-12)	73.54	90.70	89.72	80.88	75.48	75.13	45.87	65.78	64.43	81.34	66.09	87.00
FLUX.2-klein-4b (2026-01)	72.31	98.40	86.23	80.45	77.09	80.20	56.19	70.86	75.39	85.07	13.22	–
Z-Image-Turbo (2025-11)	71.40	90.00	92.25	74.57	69.30	71.57	39.68	64.57	63.02	78.36	70.69	82.00
HiDream-I1-Full (2025-05)	71.36	92.30	93.67	73.40	72.53	74.24	40.45	62.43	60.31	77.61	66.67	83.00

Table 3: **LongText-Bench.** (†) Extended multi-reward. Standard inference.

Method	English	Chinese	Overall
FLUX.1-dev (Step = 30)	0.607	0.005	0.612
+ Flow-GRPO-S3 (w/ tree)	0.669	0.026	0.695
+ Flow-GRPO-S3 (w/ tree&Trun.)	0.680	0.023	0.703
+ DFS-GRPO-S3 (Truncate)	0.696	0.035	0.731
+ DFS-GRPO-S3 (Multi-Reward)†	**0.718**	**0.036**	**0.754**
Independent models (for reference)			
HiDream-I1-Full	0.543	0.024	0.567
Kolors 2.0	0.258	0.329	0.587
Seedream 3.0	0.896	0.878	1.774
GPT-4o	0.956	0.619	1.575

Table 4: **OneIG-Bench Alignment.** (†) Extended multi-reward, standard inference.

Method	Alignment ↑
Open-source models	
FLUX.1-dev (T5)	0.786
SD 3.5 Large	0.809
HiDream-I1-Full	0.829
Z-Image-Turbo (VLM)	0.840
Closed-source models	
Imagen3	0.843
GPT-4o	0.851
DFS-GRPO (Multi-Reward)†	**0.8407**

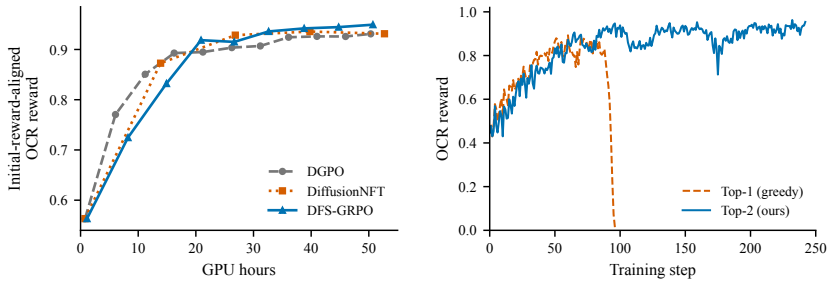
improvements of 2.4%, 5.1%, **and** 5.6% over Flow-GRPO-S3 with tree sampling and truncation. The expanded recipe strengthens fine-grained instruction following: UniGenBench text rendering rises from 43.97 to 52.30, and attribute accuracy from 78.95 to 80.02 (Table 2).

Competitive with newer-generation image models. Post-training brings FLUX.1-dev into the performance range of newer models (Tables 2 and 4). DFS-GRPO scores 72.81 on UniGenBench, compared with 72.31 for FLUX.2-klein-4b and 71.40 for Z-Image-Turbo. On OneIG-Bench Alignment, it raises FLUX.1-dev from 0.786 to 0.8407, comparable to Z-Image-Turbo (0.840) and Imagen 3 (0.843). Thus, reward-guided post-training substantially strengthens an existing backbone while retaining standard inference.

5.2 Efficient Optimization with Continued Exploration

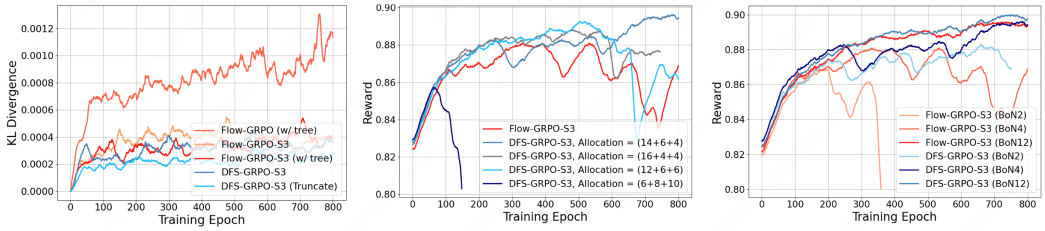
Competitive efficiency with forward-process optimization. DFS-GRPO achieves competitive OCR reward gains against DiffusionNFT (Zheng et al., 2025) and DGPO (Luo et al., 2025) at comparable GPU budgets on SD3-Medium, using the same Flow-GRPO OCR reward and prompts (Figure 5a). Whereas these baselines construct forward-noising objectives from online samples, DFS-GRPO optimizes reverse denoising trajectories with reward-guided sampling. After slower initial progress, DFS-GRPO continues improving as the baselines plateau, gaining 0.386 at the final checkpoints (50–53 GPU hours), versus approximately 0.368 for each baseline. This is consistent with positive feedback between reward-guided sampling and policy improvement (Sec. 4). Curves are aligned at initialization to compare reward gains; alignment and compute accounting appear in Appendix D.2.

Retaining alternative branches sustains learning. Under the same rollout budget, fully greedy Top-1 selection concentrates search on a single prefix and eventually collapses after rapid early gains, whereas Top-2 sustains stable reward improvement (Figure 5b).



(a) OCR optimization versus GPU hours. (b) Retaining exploration branches.

Figure 5: **Efficient reward optimization and sustained exploration on SD3-Medium.** (a) Compared with the fast forward-process methods DiffusionNFT and DGPO, DFS-GRPO improves more slowly at first but achieves larger reward gains at comparable GPU budgets, consistent with the positive feedback between reward-guided sampling and policy improvement. (b) Fully greedy Top-1 selection narrows exploration: rapid early gains give way to instability and collapse. Top-2 retains diverse continuations and sustains stable reward improvement.



(a) KL divergence (BoN4) (b) Allocation for DFS-GRPO (BoN4) (c) Different BoN Parameters

Figure 6: **Exploitation and exploration under a fixed rollout budget.** On SD3-Medium, (a) initial noise truncation keeps KL divergence low; (b) allocating $(14, 6, 4)$ rollouts balances broad initial exploration with focused prefix expansion and sustains reward improvement; (c) DFS-GRPO with BoN 4 achieves rewards close to BoN 12 while selecting fewer rollouts for optimization.

Allocating the rollout budget. With 24 rollouts, $(14, 6, 4)$ balances broad early exploration with focused expansion and yields high, stable rewards (Figure 6(b)). Initial noise truncation strengthens exploitation while keeping KL divergence from the reference model low (panel (a)). BoN 4 achieves rewards close to BoN 12 (panel (c)); we use $N = 4$ for optimization.

Extension to video generation. The same sampling strategy also improves Wanx2.1-1.3B with per-frame PickScore rewards. DFS-GRPO with BoN 4 reaches 20.81, compared with 20.68 for Flow-GRPO-S3 with tree sampling and BoN 4. The training curves and full timing comparison are given in Figure 12 and Table 6.

6 Conclusion

DFS-GRPO directs a fixed rollout budget toward high-reward regions while preserving exploration for continued learning. Its core multi-tree DFS sampler reuses promising denoising prefixes, with initial noise truncation strengthening exploitation and sub-tree budget allocation and Top-2 retention maintaining exploration. Our local analysis explains how reward-based selection directs sampling toward higher expected reward, and the Gaussian-mixture analysis connects this direction to high-reward confidence. Experiments across multiple T2I backbones and rewards demonstrate improved optimization efficiency and stronger generative performance. We also extend DFS-GRPO algorithms to the T2V task and show the potential of DFS-GRPO.

Future Work and Limitations. For DFS-GRPO, though we performed hierarchical sampling, we calculate the advantage function uniformly. It will be interesting to calculate the advantage in the sub-trees to provide a more accurate gradient. For T2V generation, we still adopt an image reward for fast training. An interesting future work is to use video reward models such as VideoAlign (Liu et al., 2025b) to improve the generative results with DFS-GRPO.

AI USE STATEMENT

We used generative AI tools to assist with language polishing, manuscript organization, literature review, theoretical analysis and proof refinement, and interpretation of experimental results. The authors take full responsibility for the final manuscript.

ETHICS STATEMENT

Our work aims to improve the generation quality via the GRPO post-training algorithm. With our models, we can generate high-quality deepfake images and videos. To address this concern, we can use a watermarking technique in the generation process (Lu et al., 2024).

Other societal impacts are similar to those of general generative models (Mirsky & Lee, 2021).

REPRODUCIBILITY STATEMENT

We provide the experimental settings and evaluation protocols in Section 5, with hyperparameters and detailed algorithms in Appendix B. The assumptions underlying our theoretical results are stated in Section 4.1, and full derivations and proofs are provided in Appendix C.

REFERENCES

Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun. Reinforcement learning: Theory and algorithms. *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, 32:96, 2019.

Anonymous. Elucidating guidance in variance exploding diffusion models: Fast convergence and better diversity, 2026. URL https://openreview.net/forum?id=tWbasgJA68.

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

Arwen Bradley and Preetum Nakkiran. Classifier-free guidance is a predictor-corrector. *arXiv preprint arXiv:2408.09000*, 2024.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

Jingjing Chang, Yixiao Fang, Peng Xing, Shuhan Wu, Wei Cheng, Rui Wang, Xianfang Zeng, Gang Yu, and Hai-Bao Chen. OneIG-Bench: Omni-dimensional nuanced evaluation for image generation. *arXiv preprint arXiv:2506.07977*, 2025. URL https://arxiv.org/abs/2506.07977.

Muthu Chidambaram, Khashayar Gatmiry, Sitan Chen, Holden Lee, and Jianfeng Lu. What does guidance do? a fine-grained analysis in a simple setting. *Advances in Neural Information Processing Systems*, 37: 84968–85005, 2024.

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

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.

Xiaolong Fu, Lichen Ma, Zipeng Guo, Gaojing Zhou, Chongxiao Wang, ShiPing Dong, Shizhe Zhou, Ximan Liu, Jingling Fu, Tan Lit Sin, et al. Dynamic-treerpo: Breaking the independent trajectory bottleneck with structured sampling. *arXiv preprint arXiv:2509.23352*, 2025.

Shiran Ge, Chenyi Huang, Yuang Ai, Qihang Fan, Huaibo Huang, and Ran He. Expand and prune: Maximizing trajectory diversity for effective grpo in generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 41913–41922, 2026.

Zigang Geng, Yibing Wang, Yeyao Ma, Chen Li, Yongming Rao, Shuyang Gu, Zhao Zhong, Qinglin Lu, Han Hu, Xiaosong Zhang, et al. X-omni: Reinforcement learning makes discrete autoregressive image generative models great again. *arXiv preprint arXiv:2507.22058*, 2025.

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.

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

Xiaoxuan He, Siming Fu, Zeyue Xue, Weijie Wang, Ruizhe He, Yuming Li, Dacheng Yin, Shuai Dong, Haoyang Huang, Hongfa Wang, et al. Flash-grpo: Efficient alignment for video diffusion via one-step policy optimization. *arXiv preprint arXiv:2605.15980*, 2026.

Gen Li and Yuchen Jiao. Provable efficiency of guidance in diffusion models for general data distribution. *arXiv preprint arXiv:2505.01382*, 2025.

Hangyu Li, Xiangxiang Chu, Dingyuan Shi, and Wang Lin. Flowdreamer: Exploring high fidelity text-to-3d generation via rectified flow. *arXiv preprint arXiv:2408.05008*, 2024.

Junzhe Li, Yutao Cui, Tao Huang, Yinping Ma, Chun Fan, Miles Yang, and Zhao Zhong. Mixgrpo: Unlocking flow-based grpo efficiency with mixed ode-sde. *arXiv preprint arXiv:2507.21802*, 2025a.

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

Henglin Liu, Huijuan Huang, Jing Wang, Chang Liu, Xiu Li, and Xiangyang Ji. Diversegrpo: Mitigating mode collapse in image generation via diversity-aware grpo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1864–1873, 2026.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online rl. *arXiv preprint arXiv:2505.05470*, 2025a.

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

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

Shilin Lu, Zihan Zhou, Jiayou Lu, Yuanzhi Zhu, and Adams Wai-Kin Kong. Robust watermarking using generative priors against image editing: From benchmarking to advances. *arXiv preprint arXiv:2410.18775*, 2024.

Yihong Luo, Tianyang Hu, and Jing Tang. Reinforcing diffusion models by direct group preference optimization. *arXiv preprint arXiv:2510.08425*, 2025. URL https://arxiv.org/abs/2510.08425.

Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi Jaakkola, Xuhui Jia, et al. Inference-time scaling for diffusion models beyond scaling denoising steps. *arXiv preprint arXiv:2501.09732*, 2025.

Yisroel Mirsky and Wenke Lee. The creation and detection of deepfakes: A survey. *ACM Computing Surveys (CSUR)*, 54(1):1–41, 2021.

Saurabh Singh and Ian Fischer. Stochastic sampling from deterministic flow models. *arXiv preprint arXiv:2410.02217*, 2024.

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.

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

Meituan LongCat Team, Xunliang Cai, Qilong Huang, Zhuoliang Kang, Hongyu Li, Shijun Liang, Liya Ma, Siyu Ren, Xiaoming Wei, Rixu Xie, and Tong Zhang. Longcat-video technical report, 2025. URL https://arxiv.org/abs/2510.22200.

UniGenBench Team. UniGenBench leaderboard. https://huggingface.co/spaces/CodeGoat24/UniGenBench_Leaderboard, 2026.

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*, pp. 8228–8238, 2024.

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

Yibin Wang, Zhimin Li, Yuhang Zang, Yujie Zhou, Jiazi Bu, Chunyu Wang, Qinglin Lu, Cheng Jin, and Jiaqi Wang. Pref-grpo: Pairwise preference reward-based grpo for stable text-to-image reinforcement learning. *arXiv preprint arXiv:2508.20751*, 2025b.

Yuchen Wu, Minshuo Chen, Zihao Li, Mengdi Wang, and Yuting Wei. Theoretical insights for diffusion guidance: A case study for gaussian mixture models. *arXiv preprint arXiv:2403.01639*, 2024.

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

Benjamin Yu, Jackie Liu, and Justin Cui. Smart-grpo: Smartly sampling noise for efficient rl of flow-matching models. *arXiv preprint arXiv:2510.02654*, 2025.

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. Diffusionnft: Online diffusion reinforcement with forward process. *arXiv preprint arXiv:2509.16117*, 2025.

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

Yujie Zhou, Pengyang Ling, Jiazi Bu, Yibin Wang, Yuhang Zang, Jiaqi Wang, Li Niu, and Guangtao Zhai. g^2 rpo: Granular grpo for precise reward in flow models. *arXiv preprint arXiv:2510.01982*, 2025.

APPENDIX

A ADDITIONAL VISUALIZATION RESULTS

In this part, we provide more examples (Figure 7 and Figure 8) to show the advantage of DFS-GRPO.

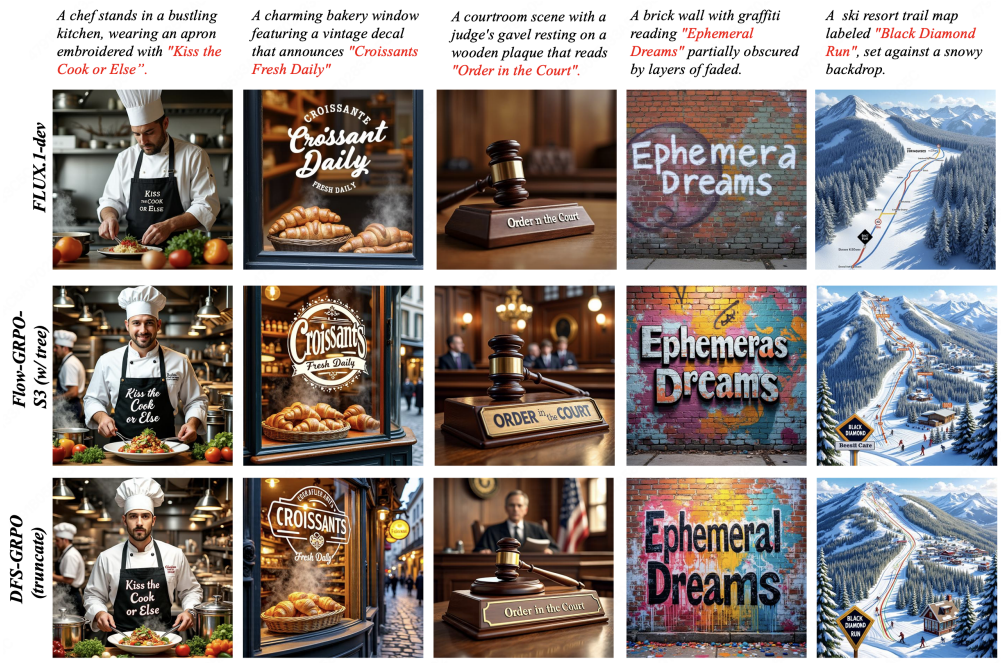


Figure 7: More Qualitative Comparison.

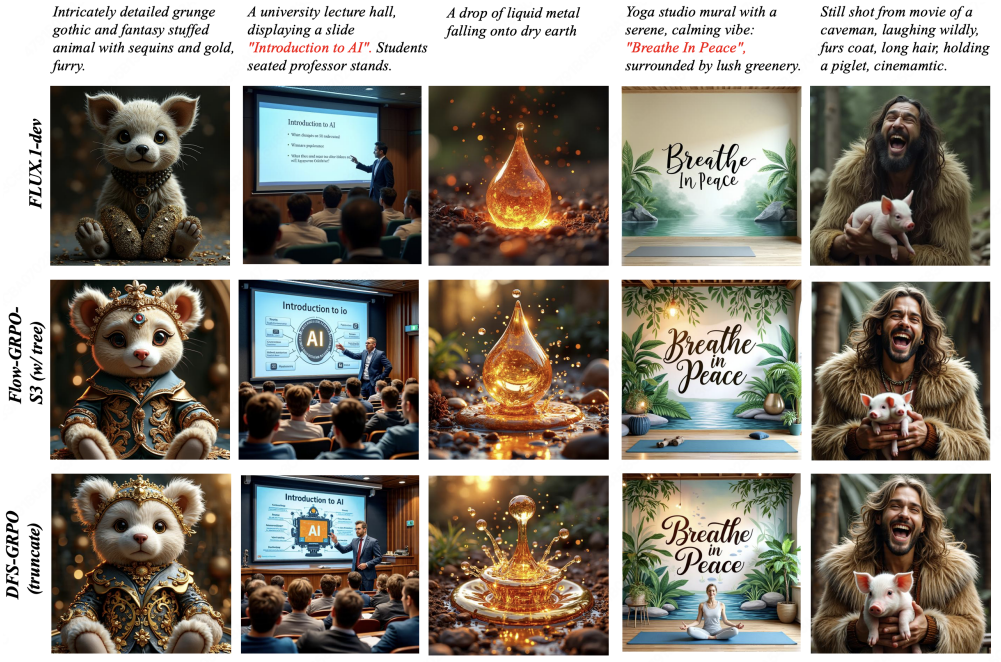
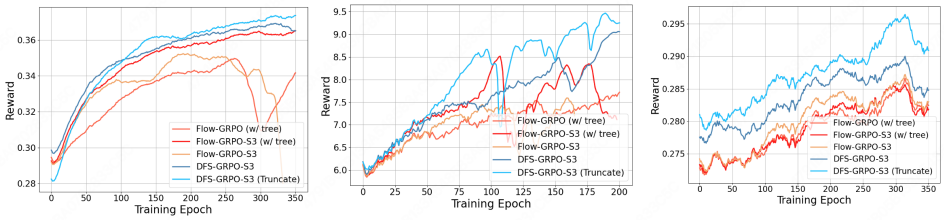


Figure 8: More Qualitative Comparison.

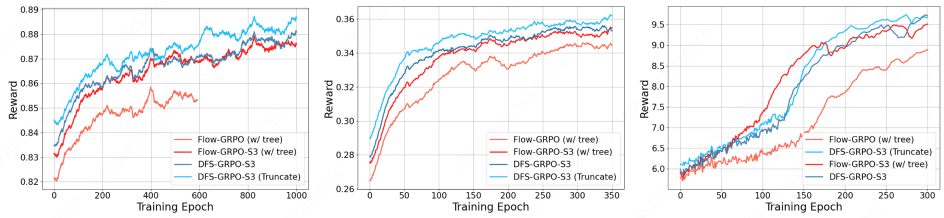
A.1 Additional Experiments

Additional Training Trajectory. The main content mainly shows the training trajectory for SD3-Medium base models on Pickscore. In this part, we show the trajectory of SD3-Medium base models (Figure 9) and FLUX.1-dev (Figure 10) with other reward models, where DFS-GRPO is still better than Flow-GRPO.



(a) HPS-v2.1 (b) Aesthetic (c) ClipScore

Figure 9: The Training Curve of DFS-GRPO on Different Reward Models (with SD3-Medium base model, BoN4). The DFS-GRPO (blue line) has a more stable and faster training process and converges to a higher reward domain compared with the baseline.



(a) Pickscore (b) HPS-v2.1 (c) Aesthetic

Figure 10: The Training Curve of Flux.1-Dev.

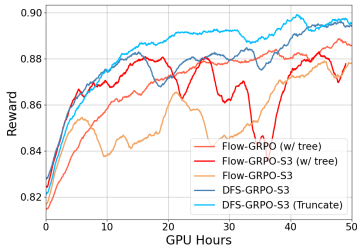


Figure 11: Trajectory w.r.t. GPU hours.

The Training Trajectory w.r.t the GPU Hours. As the above training trajectory mainly focuses on the reward trajectory w.r.t. epoch and the DFS-GRPO algorithm requires a slightly longer time per-iteration, we report the reward trajectroy w.r.t. the GPU hours. As shown in Figure 11, the DFS-GRPO still performs better than Flow-GRPO when considering the GPU hours

Repeated Experiments and Stability In this part, with BoN4, we run multiple experiments on the Flow-GRPO-S3 and DFS-GRPO-S3, and show that Flow-GRPO-S3 is unstable. As a result, we report the best one of Flow-GRPO-S3 in our main content, which is still significantly worse compared to DFS-GRPO-S3 (Figure 1).

B Experiments Detail and Hyperparameters

B.1 Hyperparameters

For the SD3-Medium base model, the number of steps is 10 in the GRPO training and 40 in the evaluation. For ODE-SDE-ODE sampling, we randomly choose the SDE starting step k_{SDE} from

$k \in [0, 5]$ for each prompt. The batch contains 32 prompts, and the group size for each prompt is 24. The learning rate is $3e-4$ and the truncated level is 1.5. The KL loss weight β is 0.01 for all reward models.

For FLUX.1-dev, the number of steps is 7 in the GRPO training and 28 in the evaluation. We randomly choose the SDE starting step k_{SDE} from $k \in [0, 1]$ for each prompt. The other setting is the same compared with the setting of SD3-Medium.

For Wanx 2.1 1.3B text-to-video models, the number of frames is 17, the height is 480, and the width is 832. The number of steps is 20 in the GRPO training and 50 in the evaluation. The batch contains 24 prompts, and the group size for each prompt is 24. Similar to (Team et al., 2025), we reweight the GPRO loss and KL loss to have a normal gradient norm. More specifically, we times 30 in the GRPO loss and set $\beta = 0.2$ for the KL loss.

B.2 Initial Noise Truncation

During training, the truncation operation draws $x_0 \sim \mathcal{TN}(0, I_d; -a, a)$ with $a = 1.5$. Each coordinate z_i is sampled independently with density

$$p(z_i) = \begin{cases} \dfrac{\phi(z_i)}{\Phi(a) - \Phi(-a)}, & z_i \in [-a, a], \\ 0, & \text{otherwise}, \end{cases} \tag{7}$$

where ϕ and Φ are the density and cumulative distribution function of $\mathcal{N}(0, 1)$. The threshold controls the concentration of the starting distribution; smaller values further restrict the initial search space. Evaluation uses standard Gaussian initialization.

B.3 Algorithm

Algorithms 1 and 2 give DFS-GRPO and its sampler. In the pseudocode, n_{k_s} is the number of new rollouts per retained prefix in sub-tree k_s, while G_{k_s} in the main text is the total number in that sub-tree. With one initial prefix and Top-2 retention thereafter, $(n_1, n_2, n_3) = (14, 3, 2)$ gives $(G_1, G_2, G_3) = (14, 6, 4)$. The retention count is fixed at $m = 2$; s indexes training iterations.

Algorithm 1 DFS-GRPO Algorithm

Input: Prompt dataset $\mathcal{C}$, policy π_θ, behavior policy $\pi_{\theta_{\text{old}}}$, Reward Model R, denoised step K, SDE step number K_s, per-prefix expansion counts $\{n_{k_s}\}_{k_s=1}^{K_s}$, keep-top $m = 2$, truncation level a, learning rate η
for iteration $s = 1$ to M **do**
  $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$ ## sync behavior policy
  Sample minibatch $\mathcal{C}_b \subset \mathcal{C}$ ## prompts in this iteration
  for prompt $c \in \mathcal{C}_b$ **do**
    if Truncate Initial Pure Noise **then**
      Draw x_0 from $\mathcal{TN}(0, I_d; -a, a)$
    else
      Draw x_0 from $\mathcal{N}(0, I_d)$
    end if
    // Run DFS-Sampling for this prompt
    $\{(x_K^i, r^i)\}_{i=1}^{N_c} \leftarrow$ DFS-SAMPLING$(c, \pi_{\theta_{\text{old}}}, R, K, K_s, \{n_{k_s}\}, k_{\text{SDE}}, m, x_0)$
    Compute GRPO loss $\mathcal{J}_{\text{GRPO}}(\theta)$
  end for
  Update policy: Gradient ascent on $\mathcal{J}_{\text{GRPO}}(\theta)$
end for

Algorithm 2 DFS Sampling

Require: Prompt c, behavior policy $\pi_{\theta_{\text{old}}}$, reward $R(\cdot)$, total steps K, #SDE steps K_s, per-prefix expansion counts $\{n_{k_s}\}_{k_s=1}^{K_s}$, starting step k_{SDE}, keep-top m, pure noise x_0

for $k = 1$ to $k_{\text{SDE}} - 1$ **do**
  $x_k \leftarrow \text{PFODE}(x_{k-1}, k; \pi_{\theta_{\text{old}}}, c)$
end for
$P \leftarrow \{(x_{k_{\text{SDE}}-1}, [\,])\}$ ## prefix pool: (state, fixed noises)
$S \leftarrow [\,]$ ## all evaluated leaves
for $k_s = 1$ to K_s **do**
  $k \leftarrow k_{\text{SDE}} + k_s - 1$
  $C \leftarrow [\,]$ ## candidates at current SDE step
  for all $(\tilde{x}, \mathbf{e}) \in P$ **do**
    for $j = 1$ to n_{k_s} **do**
      Sample $\epsilon_k \sim \mathcal{N}(0, I)$
      $x \leftarrow \text{SDE}(\tilde{x}, k, \epsilon_k; \pi_{\theta_{\text{old}}}, c)$
      for $t = k + 1$ to $k_{\text{SDE}} + K_s - 1$ **do**
        Sample $\tilde{\epsilon}_t \sim \mathcal{N}(0, I)$
        $x \leftarrow \text{SDE}(x, t, \tilde{\epsilon}_t; \pi_{\theta_{\text{old}}}, c)$ ## later SDE steps; do not record into $\mathbf{e}$
      end for
      for $t = k_{\text{SDE}} + K_s$ to $K - 1$ **do**
        $x \leftarrow \text{PFODE}(x, t; \pi_{\theta_{\text{old}}}, c)$
      end for
      $r \leftarrow R(x, c)$
      append $(x, r, \mathbf{e} \oplus [\epsilon_k])$ to C ## only record the explored SDE at step k
      append x to S
    end for
  end for
  $C_{\text{top}} \leftarrow \text{Top}_m(C \text{ by } r)$
  $P \leftarrow \{(\text{ReconstructState}(k, \mathbf{e}), \mathbf{e}) : (\cdot, \cdot, \mathbf{e}) \in C_{\text{top}}\}$
end for
Concatenate leaves $\{x_K^i\}$ from S and calculate $\hat{A}_k^i$

Algorithm 3 ReconstructState

Require: Step index k, noise list $\mathbf{e} = [\epsilon_1, \epsilon_2, \ldots, \epsilon_k]$, initial state x_0, prompt c, policy $\pi_{\theta_{\text{old}}}$
Ensure: Reconstructed state x_k

$x \leftarrow x_0$
for $i = 1$ to k **do**
  if $i < k_{\text{SDE}}$ **then**
    $x \leftarrow \text{PFODE}(x, i; \pi_{\theta_{\text{old}}}, c)$ ## ODE step before stochastic expansion
  else
    $x \leftarrow \text{SDE}(x, i, \epsilon_i; \pi_{\theta_{\text{old}}}, c)$ ## apply stored noise at SDE step
  end if
end for
return x

C Analysis of Reward-Guided Search

We develop the three views in Sec. 4.1: local selection, reward confidence, and policy optimization. The local result concerns selection among stochastic candidates at a fixed denoising stage. The Gaussian-mixture model makes its reward geometry explicit, and a coupled guided-process analysis describes how reward confidence evolves during an RF trajectory.

C.1 Theoretical Guarantees for Guidance Methods

Previous theoretical work shows that additional guidance can improve classification confidence (Bradley & Nakkiran, 2024; Chidambaram et al., 2024; Li & Jiao, 2025). Quantitatively,

Wu et al. (2024) establish convergence rates with respect to guidance strength η for DDPM-based SDEs/ODEs, while Anonymous (2026) compare DDPM and EDM convergence. Our local analysis shows how reward-based selection biases the expected state update toward the gradient of expected reward. A Gaussian-mixture model of the reward-induced sample distribution connects this direction to high-reward confidence, with multi-mode posterior comparisons and a guided-surrogate comparison for RF sampling.

C.2 Proof of Local Directional Alignment

We prove Theorem 4.1 for the candidate states in Eq. (3). Fix the state x, drift b, noise amplitude $\nu > 0$, and finite candidate count n. Break ties in the selected index by taking the smallest index. Write $u = \nabla J(x)/\|\nabla J(x)\|$ and decompose each Gaussian perturbation as

$$\varepsilon_i = Z_i u + \xi_i, \qquad Z_i = \langle u, \varepsilon_i \rangle \sim \mathcal{N}(0,1), \qquad \xi_i \sim \mathcal{N}(0, I_d - uu^\top). \tag{8}$$

The vector of projections $(Z_1, \ldots, Z_n)$ is independent of all orthogonal components $(\xi_1, \ldots, \xi_n)$.

Proof of Theorem 4.1. Couple the candidates for all $h > 0$ using the same Gaussian draws. Differentiability at x gives, for every fixed draw and each candidate,

$$\begin{aligned} J(x_i^+(h)) &= J(x) + \langle \nabla J(x), bh + \nu\sqrt{h}\,\varepsilon_i \rangle + o(\sqrt{h}) \\ &= J(x) + \nu\sqrt{h}\,\|\nabla J(x)\| Z_i + o(\sqrt{h}). \end{aligned} \tag{9}$$

There are finitely many candidates, so the remainder divided by $\sqrt{h}$ converges to zero uniformly over their indices for each fixed draw. Independent Gaussian projections have an almost surely unique maximizer $I = \arg\max_{1 \le i \le n} Z_i$ with a positive gap to the runner-up. Consequently, the index selected by $J(x_i^+(h))$ equals I for all sufficiently small h, almost surely.

Selection of I depends only on the projections. Independence and centering of the orthogonal components therefore imply

$$\mathbb{E}[\varepsilon_I] = \mathbb{E}[Z_I]u + \mathbb{E}[\xi_I] = \lambda_n u, \qquad \lambda_n = \mathbb{E}\left[\max_{i \le n} Z_i\right]. \tag{10}$$

Moreover, $\|\varepsilon_{i^*(h)}\| \le \max_{i \le n} \|\varepsilon_i\|$, whose expectation is finite. Dominated convergence yields $\mathbb{E}[\varepsilon_{i^*(h)}] = \lambda_n u + o(1)$. Substitution into the candidate update proves

$$\mathbb{E}[x_{\text{sel}}^+ - x \mid x] = bh + \nu\sqrt{h}\,\lambda_n \frac{\nabla J(x)}{\|\nabla J(x)\|} + o(\sqrt{h}). \tag{11}$$

For $n \ge 2$, $\lambda_n \ge \mathbb{E}[\max(Z_1, Z_2)] = 1/\sqrt{\pi} > 0$. ■

The result uses the continuation value to isolate the direction introduced by ranking. DFS-GRPO obtains prefix scores from completed-rollout rewards and allocates additional continuations to promising prefixes. Subtree allocation and branch retention determine how this directed search is combined with exploration, as examined in Figures 5b and 6.

C.3 Reward-Induced Gaussian Mixtures

Model and noise convention. Let C be a reward-mode label with $\Pr(C = j) = w_j > 0$ and $\sum_j w_j = 1$. The completed sample $Y \in \mathbb{R}^d$ follows

$$Y \mid C = j \sim \mathcal{N}(\mu_j, \Sigma_0), \qquad p^r(y) = \sum_{j=1}^{M} w_j \mathcal{N}(y; \mu_j, \Sigma_0), \qquad \Sigma_0 \succ 0. \tag{12}$$

These components organize sample-space regions by reward level; $+$ denotes the target high-reward component. The density p^r is the reward-induced sample model used for post-training analysis. Under the forward noising law $X_t = m_t Y + \sigma_t Z$, with $Z \sim \mathcal{N}(0, I_d)$ independent of (C, Y),

$$p_t^r(x) = \sum_j w_j \mathcal{N}(x; m_t\mu_j, \Sigma_t), \qquad \Sigma_t = m_t^2\Sigma_0 + \sigma_t^2 I_d. \tag{13}$$

RF uses $m_t = 1 - t$ and $\sigma_t = t$, with $t = 0$ denoting the clean sample. The fixed-level means and covariance denoted by μ_j and Σ in the main text correspond here to $m_t\mu_j$ and Σ_t.

Scores, posteriors, and reward confidence. Define $P_{t,j}(x) = \Pr(C = j \mid X_t = x)$. Cancelling the common quadratic term in the Gaussian densities gives

$$P_{t,j}(x) = \frac{w_j \exp\big(m_t \mu_j^\top \Sigma_t^{-1} x - \frac{1}{2} m_t^2 \mu_j^\top \Sigma_t^{-1} \mu_j\big)}{\sum_k w_k \exp\big(m_t \mu_k^\top \Sigma_t^{-1} x - \frac{1}{2} m_t^2 \mu_k^\top \Sigma_t^{-1} \mu_k\big)}. \tag{14}$$

The component score, mixture score, and high-reward guidance are

$$\begin{aligned} s_t(x, j) &= \nabla_x \log p_t^r(x \mid C = j) = -\Sigma_t^{-1}(x - m_t \mu_j), \\ \nabla_x \log p_t^r(x) &= \sum_j P_{t,j}(x) s_t(x, j), \end{aligned} \tag{15}$$

$$\nabla_x \log P_{t,+}(x) = m_t \Sigma_t^{-1} \left(\mu_+ - \sum_j P_{t,j}(x) \mu_j \right). \tag{16}$$

The clean-sample reward confidence is $P_{0,+}(y)$; its counterpart at noise level t is $P_{t,+}(x)$. These closed forms expose the connection between mixture geometry and guidance (Wu et al., 2024).

Connecting confidence to continuation value. For the model reward $R_{\mathrm{mix}}(Y) = P_{0,+}(Y)$, continue from x by drawing Y from its model conditional distribution given $X_t = x$, as the exact time-reversal SDE with $\alpha \equiv 1$ does. The Markov relation $C \to Y \to X_t$ then gives

$$\begin{aligned} J_t^{\mathrm{mix}}(x) &= \mathbb{E}[P_{0,+}(Y) \mid X_t = x] \\ &= \mathbb{E}[\mathbb{E}[\mathbf{1}\{C = +\} \mid Y, X_t] \mid X_t = x] = P_{t,+}(x). \end{aligned} \tag{17}$$

For two modes, let $a_t = m_t \Sigma_t^{-1}(\mu_+ - \mu_-)$. Differentiating the posterior gives

$$\nabla J_t^{\mathrm{mix}}(x) = P_{t,+}(x)(1 - P_{t,+}(x)) a_t, \qquad \nabla \log P_{t,+}(x) = (1 - P_{t,+}(x)) a_t. \tag{18}$$

Whenever $a_t \neq 0$, both directions coincide with the covariance-scaled low-to-high mean direction. Theorem 4.1 therefore connects value-based selection to reward guidance in this model.

Log-odds improvement from local selection. For the two-mode model, the log posterior odds $\ell_t(x) = \log(P_{t,+}(x)/P_{t,-}(x))$ are affine in x, with gradient a_t. Let x^+_{rand} be a uniformly selected candidate, independently of its reward. Theorem 4.1 and Eq. (18) yield, at the fixed noise level,

$$\mathbb{E}[\ell_t(x^+_{\mathrm{sel}}) - \ell_t(x^+_{\mathrm{rand}}) \mid x] = \nu \sqrt{h}\, \lambda_n \|a_t\| + o(\sqrt{h}). \tag{19}$$

This describes the expected log-odds advantage contributed by selection over a random candidate. The following comparison quantifies the confidence change for a given state displacement.

C.4 Confidence Comparison for Two and Multiple Modes

Fix a noise level and suppress its index. Write a_j for the component means at this level and $C_0 \succ 0$ for their common covariance. For two states x, x', define the target-versus-competitor gains

$$\Delta_j = (a_+ - a_j)^\top C_0^{-1}(x' - x) = \log \frac{P_+(x')}{P_j(x')} - \log \frac{P_+(x)}{P_j(x)}, \qquad \Delta_{\mathrm{odds}} = \min_{j \neq +} \Delta_j. \tag{20}$$

Proposition C.1 (Reward-confidence comparison)**.** *For a shared-covariance Gaussian mixture with positive weights, if $\Delta_{\mathrm{odds}} \geq 0$, then*

$$P_+(x') \geq \frac{P_+(x)}{P_+(x) + (1 - P_+(x)) e^{-\Delta_{\mathrm{odds}}}} \geq P_+(x). \tag{21}$$

The first inequality is an equality in the two-mode case.

Proof. The posterior odds of competitor j against the target are

$$\rho_j(x) = \frac{P_j(x)}{P_+(x)} = \frac{w_j}{w_+} \exp\big((a_j - a_+)^\top C_0^{-1} x - \tfrac{1}{2}(a_j^\top C_0^{-1} a_j - a_+^\top C_0^{-1} a_+)\big). \tag{22}$$

Equation (20) gives $\rho_j(x') = \rho_j(x)e^{-\Delta_j} \le \rho_j(x)e^{-\Delta_{\text{odds}}}$. Since $P_+(x) = (1 + \sum_{j\neq+} \rho_j(x))^{-1}$,

$$P_+(x') \ge \frac{1}{1 + e^{-\Delta_{\text{odds}}} \sum_{j\neq+} \rho_j(x)} = \frac{P_+(x)}{P_+(x) + (1 - P_+(x))e^{-\Delta_{\text{odds}}}}. \tag{23}$$

Nonnegativity of Δ_{odds} gives the second inequality. With only one competitor, the minimum is its actual gain and the first relation is exact. ■

This comparison extends the confidence formula to an arbitrary number of reward modes. Its geometric condition concerns the displacement's projections onto the target-versus-competitor directions. The local-selection theorem concerns the expected direction of a selected candidate; the comparison above converts specified pairwise log-odds gains into confidence.

C.5 A Guided-Process Comparison for Rectified Flow

We now study the evolution of reward confidence in a continuous guided model, following the guidance viewpoint of Wu et al. (2024). Consider the two-mode model with $\Sigma_0 = I_d$. Use reverse time u, so the forward noise level is $t = 1 - u$, and write

$$v(u) = u^2 + (1-u)^2, \qquad \kappa(u) = \frac{1 + \alpha(u)^2}{2} g(1-u)^2, \qquad g(1-u)^2 = \frac{2(1-u)}{u}. \tag{24}$$

The schedule $\alpha(u) \in [0, 1]$ selects ODE or SDE segments. On $[u_a, u_b] \subset (0, 1]$, compare a conditional process and a process with additional nonnegative guidance $\eta(u)$, driven by the same Brownian motion:

$$\begin{aligned} \mathrm{d}X^c_u &= \left[\frac{X^c_u}{u} + \kappa(u) s_{1-u}(X^c_u, +)\right] \mathrm{d}u + \alpha(u) g(1-u)\, \mathrm{d}B_u, \\ \mathrm{d}X^g_u &= \left[\frac{X^g_u}{u} + \kappa(u) s_{1-u}(X^g_u, +) + \eta(u)\kappa(u) \nabla \log P_{1-u,+}(X^g_u)\right] \mathrm{d}u \\ &\quad + \alpha(u) g(1-u)\, \mathrm{d}B_u. \end{aligned} \tag{25}$$

Assume bounded piecewise-continuous schedules α and η on this interval. Let $d = \mu_+ - \mu_-$, $H(u) = d^\top (X^g_u - X^c_u)$, and

$$A(u) = \frac{1}{u} - \frac{\kappa(u)}{v(u)}, \qquad Q(u) = \frac{\kappa(u) u}{v(u)}, \qquad \Phi(u) = \exp\left(\int_{u_a}^{u} A(s)\, \mathrm{d}s\right). \tag{26}$$

Proposition C.2 (Confidence ordering for the guided RF model)**.** *If $H(u_a) \ge 0$, then $H(u) \ge 0$ almost surely for every $u \in [u_a, u_b]$. Writing $\Delta(u) = uH(u)/v(u)$, the reward confidences satisfy*

$$P_{1-u,+}(X^g_u) = \frac{P_{1-u,+}(X^c_u)}{P_{1-u,+}(X^c_u) + (1 - P_{1-u,+}(X^c_u))e^{-\Delta(u)}} \ge P_{1-u,+}(X^c_u). \tag{27}$$

Proof. The conditional score is $s_{1-u}(x, +) = -(x - u\mu_+)/v(u)$, while $\nabla \log P_{1-u,+}(x) = u(1 - P_{1-u,+}(x))d/v(u)$. Subtracting the coupled equations cancels the Brownian increments and the common mean term. Thus the projected difference is absolutely continuous and obeys

$$H'(u) = A(u)H(u) + \eta(u)Q(u)\|d\|^2(1 - P_{1-u,+}(X^g_u)) \tag{28}$$

for almost every u. Variation of constants gives the exact identity

$$H(u) = \Phi(u)\left[H(u_a) + \|d\|^2 \int_{u_a}^{u} \frac{\eta(s)Q(s)}{\Phi(s)}(1 - P_{1-s,+}(X^g_s))\, \mathrm{d}s\right]. \tag{29}$$

Every term in brackets is nonnegative and $\Phi(u) > 0$, proving the ordering. At forward noise level $1 - u$, the high-versus-low log-odds difference is $u\, d^\top (X^g_u - X^c_u)/v(u) = \Delta(u)$. The two-mode identity in Proposition C.1 then yields Eq. (27). ■

Equation (29) describes how the confidence advantage accumulates while guidance is active. On a subsequent segment with $\eta = 0$, the projected difference evolves by a positive integrating factor, preserving the ordering through the remaining conditional ODE or SDE steps. The local selection result and this guided model give complementary views of the same reward direction: selection contributes an expected noise bias, and explicit guidance contributes a drift along the high-reward posterior gradient.

C.6 POLICY OPTIMIZATION

Return to the training reward R and the policy objective $V(\theta) = \mathbb{E}_{\pi_\theta}[R(Y)]$, averaging over the initial noise and the subsequent denoising trajectory. For a fixed policy, $J(x)$ describes the expected reward of a continuation from a particular state. In contrast, V describes how the expected reward changes as the policy parameters change. We give the smooth-ascent argument underlying Eq. (6), a standard optimization bound (Agarwal et al., 2019).

Lemma C.3 (Smooth-ascent bound)**.** *Let $V : \mathbb{R}^p \to \mathbb{R}$ have an L-Lipschitz gradient, with $L > 0$, and satisfy $V(\theta) \le V^*$. For $\theta_{s+1} = \theta_s + \nabla V(\theta_s)/L$ and any integer $S \ge 1$,*

$$\min_{0 \le s < S} \|\nabla V(\theta_s)\|^2 \le \frac{2L[V^* - V(\theta_0)]}{S}. \tag{30}$$

Proof. Gradient Lipschitzness implies

$$V(\theta') \ge V(\theta) + \langle \nabla V(\theta), \theta' - \theta \rangle - \frac{L}{2}\|\theta' - \theta\|^2. \tag{31}$$

Substituting the ascent update yields $V(\theta_{s+1}) - V(\theta_s) \ge \|\nabla V(\theta_s)\|^2/(2L)$. Summing over $s = 0, \dots, S-1$ gives

$$\sum_{s=0}^{S-1} \|\nabla V(\theta_s)\|^2 \le 2L[V(\theta_S) - V(\theta_0)] \le 2L[V^* - V(\theta_0)]. \tag{32}$$

The minimum is no greater than the average, which proves the claim. ■

The bound describes exact ascent on the policy objective. DFS-GRPO constructs the rollout groups used by the group-relative update: promising prefixes focus sampling on high-reward continuations, while alternative branches and lower-reward rollouts provide exploration and reward contrasts. The training trajectories and compute comparisons in Figures 3, 5a and 11 characterize the resulting learning behavior.

D Additional Benchmark Results and Experimental Analysis

D.1 Full Benchmark Results

Table 2 reports every UniGenBench category and the GenEval task-averaged overall score. The extended reward recipe combines HPS-v2.1, CLIPScore, OCR, and GroundingDINO spatial rewards with structured OCR and spatial prompts. FLUX.1-dev benchmark evaluation uses standard single-LoRA inference with 30 steps and guidance scale 3.5. UniGenBench uses Gemini-2.5-Pro as its evaluator, and Table 3 reports both language subsets of LongText-Bench.

The independent-model comparisons use the reported benchmark scores from UniGenBench, LongText-Bench, and OneIG-Bench (UniGenBench Team, 2026; Geng et al., 2025; Chang et al., 2025). The main-text OneIG-Bench comparison reports its Alignment dimension. UniGenBench values follow the benchmark evaluation records, and GenEval overall is the average over its tasks.

D.2 OCR Reward Trajectories

Table 5 gives the original endpoints underlying Figure 5a. For method m at GPU time t, the plotted reward is $\widetilde{R}_m(t) = R_m(t) - R_m(0) + R_{\mathrm{DFS}}(0)$. Each curve therefore subtracts its own initial reward and adds the initial DFS-GRPO reward, making the vertical change equal to the reward gain. GPU hours equal elapsed training hours multiplied by four GPUs. The comparison uses the same SD3-Medium backbone, OCR reward, and Flow-GRPO OCR prompt sets. Endpoint GPU budgets follow the recorded evaluation checkpoints.

Table 5: **OCR reward endpoints on SD3-Medium.** Rewards in this table are the original values. Gain is final minus initial reward; GPU hours include the four GPUs used in each run.

Method	GPU hours	Initial	Final	Gain
DGPO	50.33	0.5903	0.9581	0.3678
DiffusionNFT	52.72	0.5918	0.9599	0.3680
DFS-GRPO	50.71	0.5633	0.9493	0.3861

The branch-retention ablation in Figure 5b uses the same 24-sample budget per iteration for Top-1 and Top-2. Top-2 reaches an OCR reward of 0.954 at step 242. Top-1 reaches 0.893 at step 60, then falls to 0.014 by step 96.

D.3 TEXT-TO-VIDEO GENERATION

We apply DFS-GRPO to Wanx2.1-1.3B using per-frame PickScore rewards, following the image-reward approach used by Team et al. (2025). Figure 12 and Table 6 report the training trajectories, iteration times, and evaluation scores. With BoN 4, DFS-GRPO reaches a PickScore of 20.81 and retains its reward advantage when the optimization selection budget is increased to BoN 8.

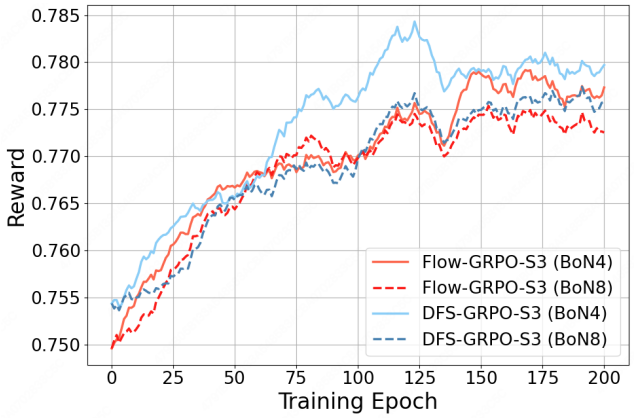


Figure 12: Training curves for text-to-video generation with Wanx2.1-1.3B and per-frame PickScore rewards.

Table 6: **Quantitative results** for text-to-video generation.

Method	Iteration Time (s) ↓	PickScore ↑
Wanx2.1-1.3B (Step = 50)	/	20.15
+ Flow-GRPO-S3 (w/ tree) (BoN4)	721.15	20.68
+ Flow-GRPO-S3 (w/ tree) (BoN8)	910.22	20.71
+ DFS-GRPO-S3 (BoN4)	769.59	**20.81**
+ DFS-GRPO-S3 (BoN8)	949.86	<u>20.75</u>