Expand and Prune: Maximizing Trajectory Diversity for Effective GRPO in Generative Models

Supplementary Material

A. Experimental Setting Details

A.1. Backbones and Baselines

- **Diffusion (SD-v1.4).** We use the official SD-v1.4 latent diffusion checkpoint (512×512) and benchmark against **DanceGRPO** as our baseline, adopting its public configuration (group size G=16, denoising steps T=50) with the default VAE and sampler.
- **Rectified Flow (SD-3.5-M).** We use the official SD-3.5-Medium checkpoint (512×512) and benchmark against **Flow-GRPO**, adopting its public configuration ($G = 24, T = 10$) with the default VAE and sampler.

A.2. Evaluation Metrics

The details of evaluation metrics are as follows:

- **ImageReward**: a BLIP-based human preference reward model trained on 136k expert pairwise comparisons to evaluate alignment, visual quality, and harmlessness.
- **HPS-v2.1**: a CLIP-based model fine-tuned on 798k pairwise comparisons to quantify human preference for T2I alignment.
- **Aesthetic Score**: A CLIP-based predictor that quantifies perceived visual appeal and predicts an image's aesthetic score.
- **PickScore**: a CLIP-based model trained on the Pick-a-Pic dataset, serving as a widely used proxy for human preference in T2I evaluation.
- **GenEval**: an object-focused benchmark that evaluates compositional constraints (e.g., color, spatial) using detectors to provide interpretable fine-grained accuracy scores.

A.3. Hyperparameters Specification

We summarize the common training parameters and contrast the specific trajectory pruning mechanisms employed by Pro-GRPO and its lightweight Flash variant. Our method uses the same optimizer, clipping, KL weighting and decoding as the corresponding baseline to ensure a fair comparison, full configurations are listed in Table 7 and 8.

B. Reward Clustering Phenomenon Analysis

To better understand the reward clustering phenomenon discussed in Sec. 4.1, we analyze the empirical reward distribution induced by GRPO sampling. Concretely, on the HPSv2 benchmark we consider 300 prompts, and for each prompt we sample G=12 trajectories under the reference

Table 7. **Hyperparameter specifications for Flow-based Pro-GRPO (SD3.5-M).** The upper section lists shared optimization settings, while the lower section delineates the distinct sampling and pruning schedules ($G_{\max}, t_1, t_2$) that differentiate the Standard and Flash models.

Hyperparameter	Pro-GRPO	Pro-GRPO Flash
General Settings		
LoRA Fine-tuning	True	True
Global Noise σ_{std}	True	True
Model EMA	True	True
Image Resolution	512×512	512×512
CFG Scale	4.5	4.5
Denoising Steps T	10	10
KL Regularization	0.01	0.01
Train Batch Size	8	8
Total Prompts per Epoch	96	96
Pruning Mechanism		
Initial Sampling ($G_{\max}$)	48	24
Intermediate Sampling	24	16
Final Trajectory (K)	12	12
First Pruning Step (t_1)	5	5
Second Pruning Step (t_2)	7	7

Table 8. **Hyperparameter specifications for Diffusion-based Pro-GRPO (SDv1.4).** We detail the optimization settings and the specific pruning schedule.

Hyperparameter	Value
General Settings	
LoRA Fine-tuning	False
Total Training Epochs	300
Denoising Steps T	50
CFG Scale	5.0
Training Batch Size	4
Prompts per Batch	8
Gradient Accumulation	8
Pruning Mechanism	
Initial Sampling ($G_{\max}$)	48
Intermediate Sampling	32
Final Trajectory (K)	8
First Pruning Step (t_1)	30
Second Pruning Step (t_2)	40

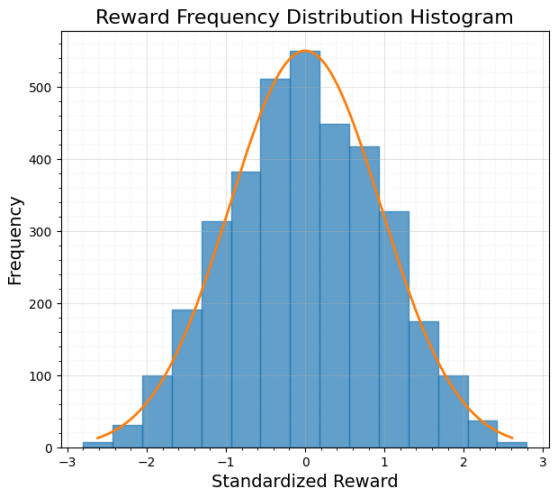

Figure 6. **Reward Distribution Analysis.** We visualize the frequency distribution of standardized rewards across 3,600 trajectories (300 prompts $\times$ 12 samples). The data closely fits a Gaussian distribution (orange curve).

policy. We computed the standardized rewards for all trajectories and visualized their frequency distribution.

As illustrated in Fig. 6, the empirical reward distribution closely approximates a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$. Under such an approximately Gaussian distribution, most probability mass concentrates near the mean. This directly explains the reward clustering phenomenon in GRPO groups: when trajectories are drawn from this distribution, random sampling naturally produces many near-mean rewards and relatively few samples in the high and low-reward tails. As a result, even if we randomly subsample trajectories from a group, the subsample still follows approximately the same Gaussian law and therefore preserves reward clustering rather than alleviating it.

Our Optimal Variance Filtering (OVF) is explicitly designed to counteract this effect. By selecting a subset that maximizes within-set reward variance, OVF preferentially retains trajectories from the two tails of the distribution instead of the dense central region. This increases the contrastive signals between rewards and strengthens the normalized advantages used for policy updates.

Furthermore, the Expand-and-Prune strategy in Pro-GRPO amplifies this effect: starting from an enlarged group size $G_{\max}$ increases the absolute number of rare tail samples available under the same underlying Gaussian, and multi-step OVF pruning then filters this expanded pool down to a high-variance survivor set. Together, this provides a principled explanation of why Pro-GRPO can break the inherent reward clustering of GRPO and yield stronger, more informative optimization signals.

Table 9. **Pilot study on pruning checkpoints selection (SD3.5-M).** Candidate schedules are compared in terms of proxy reliability, reward separability, and relative cost. We choose $(5, 7)$ as the earliest stable pruning schedule.

(t_1, t_2)	Var. ↑	Spearman ↑ (t_1 / t_2)	CostRatio ↓	Decision
(2, 4)	0.0012	0.3354 / 0.6954	0.450	Unreliable
(4, 6)	0.0017	0.6954 / 0.8579	0.600	Borderline
(5, 7)	**0.0019**	**0.7922 / 0.8957**	**0.675**	**Selected**
(7, 9)	0.0021	0.8957 / 0.9984	0.825	Inefficient

Figure 7. **Evolution of one-step previews across timesteps.** Very early previews are noisy and unreliable for ranking, while semantically meaningful structure gradually emerges at later timesteps.

C. Pilot Selection of Pruning Checkpoints

A critical design choice in Pro-GRPO is determining when to activate trajectory pruning. Applying OVF too early yields noisy, unreliable rankings, while pruning too late compromises computational savings. To resolve this trade-off, we adopt a lightweight, inference-only pilot procedure to evaluate candidate pruning pairs (t_1, t_2) along three axes: (1) *Rank Reliability*, measured by the Spearman correlation between the one-step proxy reward and the terminal reward; (2) *Reward Separability*, measured by the reward variance of the retained subset after OVF. We also record a relative cost ratio to reflect the amount of computation retained under each pruning schedule.

Table 9 summarizes the pilot results on SD3.5-M. Very early pruning (e.g., steps 2 and 4) exhibits low rank correlation, indicating the one-step preview is too unstable for trustworthy selection. Moving deeper substantially improves fidelity. We select $(5, 7)$ as our final configuration because it represents the *earliest stable* schedule—achieving strong rank correlation and reward contrast without sacrificing the massive efficiency gains lost by later schedules like $(7, 9)$. This quantitative decision aligns seamlessly with the visual evolution in Figure 7, where recognizable semantic structures only begin to emerge around step 5.

While absolute pruning steps depend on the underlying architecture and total denoising length T, we identified a highly stable operating region when normalized by T.

Across our experiments, the optimal first checkpoint consistently falls within $t_1 \in [0.5, 0.6]T$, and the second within $t_2 \in [0.7, 0.8]T$. We emphasize that this pilot study aims to identify a reliable operating region in which the one-step preview is sufficiently aligned with terminal rewards while still preserving the efficiency gains brought by early trajectory termination. In practice, this selection strategy was stable and easy to apply across the models considered in this work.

D. Efficiency Analysis

We provide a simple FLOPs model and a wall-clock time breakdown for GRPO-style training, and explain why Pro-GRPO lowers both theoretical compute and practical runtime.

We denote the FLOPs cost of atomic operations as $\mathcal{C}_{\text{unet}}$ (noise prediction), $\mathcal{C}_{\text{vae}}$ (VAE decoding), and $\mathcal{C}_{\text{rwd}}$ (reward computation). Let N be the initial group size ($G_{\max}$) and T be the total denoising steps.

D.1. Baseline GRPO

Standard methods process all N trajectories through the full pipeline. Each trajectory is denoised for T steps by the UNet (cost $\mathcal{C}_{\text{unet}}$ per step), decoded once by the VAE ($\mathcal{C}_{\text{vae}}$), and evaluated once by the reward model ($\mathcal{C}_{\text{rwd}}$). For the optimization stage, we assume a KL-regularized objective that evaluates both the current and a reference policy, and we empirically estimate the backward pass to be roughly twice as expensive as the forward pass. Together, this results in about four UNet-equivalent calls per denoising step. Summing these three stages over all trajectories yields the total baseline cost $\mathcal{T}_{\text{base}}$ in Eq. (21), which scales linearly with both the group size N and the number of denoising steps T.

$$\begin{aligned}\mathcal{T}_{\text{base}} &= \underbrace{N \cdot T \cdot \mathcal{C}_{\text{unet}}}_{\text{Sampling}} + \underbrace{N \cdot (\mathcal{C}_{\text{vae}} + \mathcal{C}_{\text{rwd}})}_{\text{Eval}} + \underbrace{N \cdot T \cdot 4\mathcal{C}_{\text{unet}}}_{\text{Optimization}} \\ &= N \cdot [5T\mathcal{C}_{\text{unet}} + \mathcal{C}_{\text{vae}} + \mathcal{C}_{\text{rwd}}] .\end{aligned} \tag{21}$$

D.2. Pro-GRPO

Pro-GRPO introduces dynamic pruning at two checkpoints t_1 and t_2. Let N_0, N_1, N_2 denote the number of active trajectories in the three resulting phases ($N_0 \to N_1 \to N_2$, with $N_0 = N$ and $N_2 = K$). The total cost $\mathcal{T}_{\text{pro}}$ in Eq. (22) explicitly separates the cost of expansion and pruning from that of denoising and optimization.

The first term accumulates the standard UNet denoising cost over the three phases, where Δt_k denotes the number of denoising steps executed in phase k. The second term captures the extra work at checkpoints t_1 and t_2: at each checkpoint we perform one additional UNet evaluation, one VAE decode, and one reward evaluation on all currently active trajectories in order to rank and prune them. The third term corresponds to the final decode and reward computation on the survivor set of size N_2. The last term mirrors the baseline optimization cost, but now only the N_2 surviving trajectories participate in policy updates.

$$\begin{aligned}\mathcal{T}_{\text{pro}} = &\underbrace{\sum_{k=0}^{2} N_k \cdot \Delta t_k \cdot \mathcal{C}_{\text{unet}}}_{\text{Denoising Stages}} + \underbrace{\sum_{k=0}^{1} N_k \cdot (\mathcal{C}_{\text{unet}} + \mathcal{C}_{\text{vae}} + \mathcal{C}_{\text{rwd}})}_{\text{Pruning Overhead}} \\ &+ \underbrace{N_2 \cdot (\mathcal{C}_{\text{vae}} + \mathcal{C}_{\text{rwd}})}_{\text{Final Eval}} + \underbrace{N_2 \cdot T \cdot 4\mathcal{C}_{\text{unet}}}_{\text{Final Optimization}}\end{aligned} \tag{22}$$

The core efficiency gain stems from limiting the dominant optimization phase (driven by $4T\,\mathcal{C}_{\text{unet}}$) to only $N_2 \ll N$ survivors. This backend saving outweighs the pruning overhead, consistent with the net FLOPs reduction reported in Sec. D.3.

D.3. Comparative FLOPs Analysis

Tables 10 and 11 report the estimated per-epoch FLOPs for Pro-GRPO and its corresponding baselines on both backbones. In all cases, Pro-GRPO achieves a consistent efficiency gain over the underlying GRPO variant: on

Table 10. **Computational efficiency on SD3.5-M (flow-based).** Per-epoch FLOPs (in tera-FLOPs) and relative speedup for Flow-GRPO and our Pro-GRPO variants on the SD3.5-M backbone.

Component / Method	FLOPs (T) ↓	Speedup ↑
Atomic Operations (Per Call)		
Noise Prediction	3.88	–
VAE Decoding	2.49	–
Reward Computation	0.34	–
Full Training Framework		
Flow-GRPO (Baseline)	453474.18	1.00×
Pro-GRPO (Standard)	335626.82	**1.26**×
Pro-GRPO (Flash)	**267365.79**	**1.41**×

Table 11. **Computational efficiency on SD-v1.4 (diffusion-based).** Per-epoch FLOPs (in tera-FLOPs) and relative speedup for DanceGRPO and Pro-GRPO on the SD-v1.4 backbone.

Component / Method	FLOPs (T) ↓	Speedup ↑
Atomic Operations (Per Call)		
Noise Prediction	1.36	–
VAE Decoding	2.49	–
Reward Computation	0.37	–
Full Training Framework		
DanceGRPO (Baseline)	21943.04	1.00×
Pro-GRPO	19,937.92	**1.09**×

Table 12. **T2I-CompBench++ Result.** This evaluation utilizes the same models presented in the main paper: the flow-based model (SD3.5-M) trained on PickScore and the diffusion-based model (SDv1.4) trained on HPSv2.1.

Model	T2I-CompBench Score (↑)			
	Color	Shape	Spatial	Texture
SDv1.4 (Base)	0.370	0.364	0.116	0.418
DanceGRPO	0.552	0.503	0.191	0.582
Pro-GRPO (Ours)	**0.622**	**0.536**	**0.198**	**0.624**
SD3.5-M (Base)	0.801	0.588	0.299	0.728
Flow-GRPO	0.829	0.641	0.359	0.746
Pro-GRPO (Ours)	**0.853**	**0.664**	**0.389**	**0.757**

Table 13. Wall-clock time breakdown per training iteration. We separately report the cumulative VAE decoding time and the cumulative latent-space denoising time.

Diffusion-based (SD-v1.4)			Flow-based (SD-3.5-M)		
Method	**VAE (s)**	**U-Net (s)**	**Method**	**VAE (s)**	**DiT (s)**
DanceGRPO	0.45	68.82	Flow-GRPO	0.33	16.96
Pro-GRPO	2.48	**59.23**	Pro-GRPO	0.60	**6.58**

SD3.5-M, Pro-GRPO (Standard) and Pro-GRPO-Flash reduce FLOPs by $\sim 26\%$ and $\sim 41\%$, respectively, while on SD-v1.4, Pro-GRPO still yields a $\sim 9\%$ reduction relative to DanceGRPO.

The magnitude of the speedup, however, differs across backbones. The efficiency gains are markedly larger on SD3.5-M than on SD-v1.4. We attribute this to the architectural and computational gap between the two generative models: the DiT-based denoiser in SD3.5-M has a substantially higher per-step cost than the U-Net denoiser in SD-v1.4, whereas the VAE decoding and reward evaluation costs are of similar scale. Since Pro-GRPO's primary saving comes from reducing the optimization batch size N_2 (which scales with noise prediction cost), the benefit is magnified on computationally intensive models.

D.4. Wall-clock Time Breakdown

We further report the wall-clock time per training iteration, isolating the cumulative VAE decoding cost from the dominant latent-space denoising (Table 13). For the diffusion-based setting, while intermediate proxy evaluations increase VAE time from 0.45s to 2.48s, this overhead is easily offset by the U-Net time reduction (68.82s down to 59.23s), lowering the total iteration time from 69.27s to 61.71s.

A similar trend is observed on the flow-based backbone: a slight increase in VAE time (0.33s to 0.60s) is vastly outweighed by a substantial drop in DiT time (16.96s to 6.58s), cutting the total time from 17.29s to 7.18s. Consistent with our FLOPs model in Sec. D.3, these measurements confirm that latent-space processing heavily dominates the runtime. The intermediate pixel-space decodings introduce minimal overhead, allowing Pro-GRPO to translate theoretical FLOPs reductions into tangible end-to-end wall-clock speedups.

Table 14. **Evaluation on an additional backbone.** This experiment follows the same training and evaluation protocol as in the main paper, instantiated on an additional diffusion-based model, SD v1.5, using HPSv2.1 as the optimization reward.

Model	In-Domain	Out-of-Domain		
	HPS v2.1 ↑	IR ↑	Pick ↑	HPS v2.1 ↑
SD v1.5 (Base)	–	0.0114	21.1179	0.2445
DanceGRPO	0.3748	0.8894	21.8453	0.3251
Pro-GRPO (Ours)	**0.3884**	**0.9543**	**21.8997**	**0.3386**

D.5. Conclusion: Speed with Expanded Scope

Ultimately, Pro-GRPO achieves a rare dual advantage. Even under the "Expand" setting ($G_{\max} > G_{\text{base}}$), the aggressive "Prune" mechanism ensures a net increase in training speed. Combined with the expand-and-prune strategy, this allows us to enjoy the exploration benefits of a large initial pool while simultaneously improving both computational efficiency and alignment performance.

E. Extended Experimental Results

E.1. Broader Validation on an Additional Backbone

To verify that the observed improvements are not specific to a particular pretrained model, we evaluate Pro-GRPO on an additional backbone, Stable Diffusion v1.5 (SD v1.5), comparing it against the DanceGRPO baseline under identical fine-tuning conditions.

As reported in Table 14, the advantage of Pro-GRPO transfers consistently to this additional backbone. For in-domain optimization, Pro-GRPO improves the HPS v2.1 score from 0.3748 to 0.3884. More importantly, these gains generalize robustly to out-of-domain evaluations: ImageReward increases significantly from 0.8894 to 0.9543, accompanied by consistent improvements in both PickScore and out-of-domain HPS v2.1. While both RL-tuned methods substantially surpass the zero-shot SD v1.5 base model, Pro-GRPO consistently achieves the best overall performance.

This suggests that the benefit of Pro-GRPO is largely architecture-agnostic and stems from improved trajectory selection during optimization, rather than from a particular choice of pretrained backbone.

E.2. Broader Validation on an Additional Benchmark

We further evaluate compositional reasoning on T2I-CompBench++. This suite specifically targets fine-grained attribute binding capabilities, including Color, Shape, Spatial, and Texture.

As shown in Table 12, Pro-GRPO consistently outperforms all baselines across backbones. On SDv1.4, Pro-GRPO delivers gains over the base model and DanceGRPO on every axis, with noticeable improvements in color and texture fidelity as well as spatial accuracy. On SD3.5-M, Pro-GRPO similarly surpasses Flow-GRPO across all four metrics, particularly strengthening spatial scores (0.389 vs. 0.359 for Flow-GRPO). These results indicate that our expand-and-prune training not only improves reward metrics, but also yields more reliable compositional grounding of attributes in complex prompts.

F. More Visualization Results

We provide extensive qualitative visualizations in the appendix to further assess our method. Across a wide range of prompts, Pro-GRPO produces images that are more faithful to complex textual descriptions, and noticeably less prone to artifacts than competing baselines, illustrating its advantages in both visual quality and semantic alignment.

Figure 8. Qualitative comparison between SD3.5-M, Flow-GRPO (Prompt) and Pro-GRPO on HPSv2 prompts.

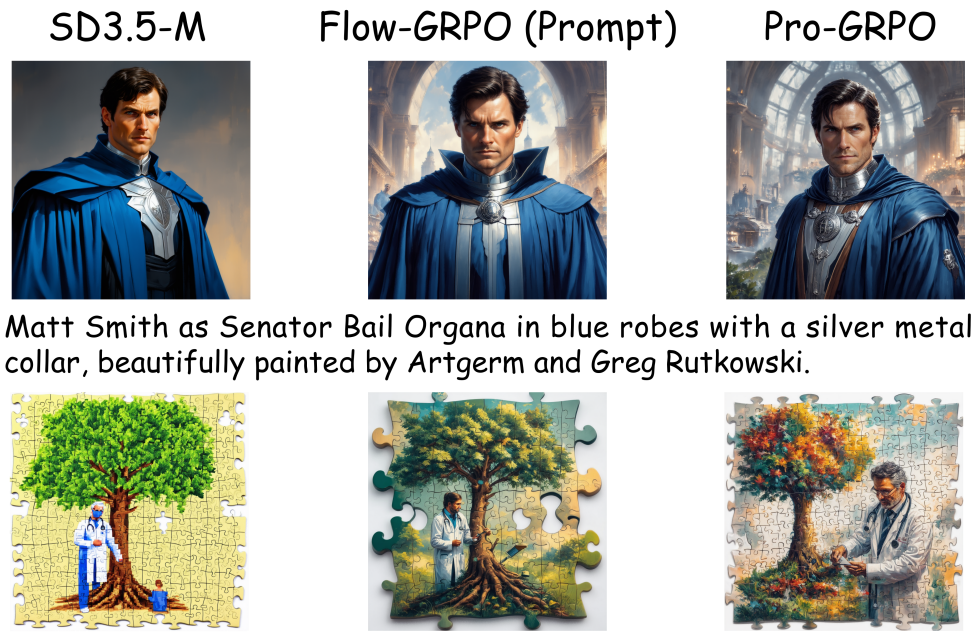

Figure 9. Qualitative comparison between SD3.5-M, Flow-GRPO (Prompt) and Pro-GRPO on HPSv2 prompts.

Figure 10. Qualitative comparison between SD3.5-M, Flow-GRPO (Prompt) and Pro-GRPO on HPSv2 prompts.

Figure 11. Qualitative comparison between SD3.5-M, Flow-GRPO (Prompt) and Pro-GRPO on HPSv2 prompts.