SMOOTH THEN BALANCE: GRADIENT SMOOTHING AND PREFERENCE ANCHORING FOR MULTI-OBJECTIVE DIFFUSION RL

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-objective diffusion reinforcement learning (RL) fine-tunes a single diffusion model to jointly improve multiple objectives (*e.g.*, image quality, human preference, and text rendering). Gradient-based multi-objective optimization methods such as the Multiple Gradient Descent Algorithm (MGDA) avoid manually designed reward weights by combining the gradients of individual rewards into a common descent direction for all rewards. However, we empirically find two drawbacks of MGDA when it is applied to diffusion RL. First, the reward gradients are highly noisy, which makes the resulting update directions unreliable. Second, MGDA determines the weights solely from the gradient directions and can yield highly imbalanced weights that leave some rewards under-optimized. To address these problems, we propose *Smooth Then Balance* (STB), which first smooths each reward gradient with an exponential moving average before the weights are computed, and then balances the smoothed gradients with a preference anchor that guarantees every reward a minimum weight. Our theoretical results prove that the error of the STB direction is bounded by the gradient estimation error, and that a small STB update guarantees closeness to Pareto stationarity once this error is small. On five-reward fine-tuning of SD3.5-Medium, STB outperforms MGDA on all eight evaluation metrics and achieves the best results on four of the five training rewards among multi-objective diffusion RL methods.

1 INTRODUCTION

A versatile text-to-image diffusion model is expected to satisfy multiple criteria at once, for example aligning with human preferences Wu et al. (2023b), following compositional instructions Huang et al. (2023), and rendering requested text correctly Chen et al. (2023a). Multi-objective reinforcement learning (RL) of diffusion models, which we refer to as *multi-objective diffusion RL*, pursues such criteria jointly by optimizing a single diffusion model against multiple rewards Zheng et al. (2026); Chen et al. (2026); Zhao et al. (2026). One common practice is to combine these rewards with fixed weights. However, the weights usually need to be manually crafted for each reward combination Lin et al. (2019), and rewards that are easier to improve may dominate the update Yu et al. (2020). Gradient-based multi-objective optimization offers an alternative that determines the update from the gradients of the individual objectives rather than from fixed weights. A representative method is the Multiple Gradient Descent Algorithm (MGDA) Désidéri (2012), which combines the per-objective gradients into a common descent direction and has been widely adopted in multi-task deep learning Sener & Koltun (2018). Recently, MARBLE Zhao et al. (2026) extended MGDA to multi-objective diffusion RL. Despite these progresses, we find that MGDA faces two problems when fine-tuning diffusion models. The reward gradients are highly noisy, and the resulting reward weights can be highly imbalanced.

The reward gradients are noisy because diffusion RL is inherently stochastic Liu et al. (2026); Zheng et al. (2026). Each reward gradient is a Monte Carlo estimate over the prompts in the batch, the initial noise from which each image is generated, and the diffusion timesteps and forward-process noise at which the training loss is evaluated Ho et al. (2020); Lipman et al. (2022); Esser et al. (2024); Black et al. (2024). Because generating and scoring images is expensive, each update can afford only a limited number of samples, so the estimate has high variance Nichol & Dhariwal (2021); Kingma

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

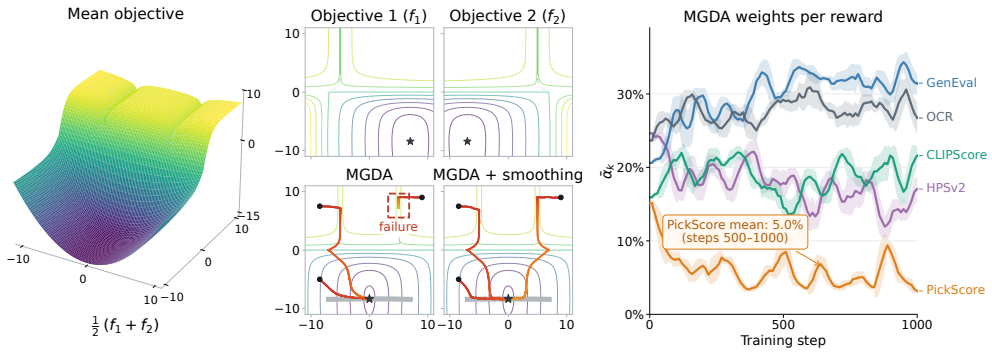


Figure 1: **Left and middle:** the toy problem from Liu et al. (2021), showing the mean objective, the contours of each objective (★: minima), and trajectories from three initializations (●) under noisy gradients, without and with objective-wise gradient smoothing. The gray curve (—) marks Pareto-stationary points; f_1 and f_2 are defined in Appendix C. **Right:** MGDA weights for five rewards during multi-objective diffusion RL, where PickScore receives persistently low weight.

et al. (2021); Xu et al. (2023b). Since MGDA computes its weights from these noisy gradients, the resulting update direction is also noisy and can be unreliable Fernando et al. (2023). Fig. 1 (left and middle) illustrates this effect on a two-objective toy problem Liu et al. (2021), where we add Gaussian noise to the gradients to mimic this setting. With noisy gradients, MGDA fails to reach the Pareto-stationary set from the upper-right initialization. In contrast, with gradient smoothing, the same run eventually reaches the Pareto-stationary set.

The imbalanced weights arise because MGDA determines the reward weights solely from the gradient directions, so a reward whose gradient conflicts with the others can receive almost no weight Liu et al. (2021). In our five-reward training, MGDA gives PickScore only 5% of the weight on average over the second half of training (Fig. 1, right), far below the 20% it would receive under uniform weighting. As a result, PickScore improves little, and MGDA raises it only from 22.34 to 22.73, below other multi-objective methods such as MapReduce LoRA (22.81) and DiffusionNFT (23.80).

To address these two problems, we propose *Smooth Then Balance* (STB), which modifies MGDA in two complementary steps. Since the noise originates from the gradient estimates, STB first maintains an exponential moving average (EMA) of each reward gradient across training rounds and computes the weights from the smoothed gradients. Smoothing alone, however, does not prevent imbalanced weights. STB therefore balances the smoothed gradients with a preference anchor adapted from SDMGrad Xiao et al. (2023), which guarantees each reward a minimum weight while the remaining weight is still learned from the gradients.

We further establish two theoretical results that clarify the role of each component of STB. First, the direction computed by STB deviates from the direction obtained with the true gradients by an amount bounded by the gradient estimation error (Proposition 3.1). This bound holds for the anchored weights used by STB and depends on the estimation error rather than on the stability of the weights. It therefore motivates the smoothing step, which reduces the estimation error whenever the reduction in noise variance outweighs the lag introduced by the EMA. Second, the distance to Pareto stationarity is bounded by the norm of the STB update plus the estimation error (Proposition 3.2). Since the update norm can be computed during training, a small update guarantees that the model is close to Pareto stationarity once smoothing has made the gradient estimates accurate. This guarantee holds for any weights that satisfy the preference anchor, so the balancing step can enforce a minimum weight for every reward without invalidating it. In summary, our main contributions are as follows:

- We identify two issues that arise when MGDA is applied to multi-objective diffusion RL. The reward gradients are too noisy to yield reliable weights, and the weights can become heavily imbalanced across rewards.
- We propose *Smooth Then Balance* (STB), which smooths each reward gradient with an EMA before the weights are computed and balances the smoothed gradients with a preference anchor that guarantees every reward a minimum weight. Our theoretical results prove that the error of the STB direction is bounded by the gradient estimation error, and that,

for any weights satisfying the anchor, a small STB update guarantees closeness to Pareto stationarity once this error is small.

- On five-reward RL fine-tuning of SD3.5-Medium, STB achieves the best scores on four of the five training rewards among multi-objective diffusion RL methods and improves over MGDA on all eight evaluation metrics.

2 Related Work

Multi-objective diffusion RL. Reinforcement learning has become a common way to align text-to-image diffusion models with human preferences and task-specific criteria. Early work fine-tunes diffusion models on a single reward, either with policy gradients Black et al. (2024); Fan et al. (2023), by backpropagating through a differentiable reward model Xu et al. (2023a); Clark et al. (2024), or from preference pairs Wallace et al. (2024). Flow-GRPO Liu et al. (2026) and DanceGRPO Xue et al. (2025) later brought GRPO to flow-matching models, and DiffusionNFT Zheng et al. (2026) moved policy optimization to the forward process. Handling several rewards is less settled. Parrot Lee et al. (2024) picks Pareto-optimal samples in each batch. Rewarded soups Rame et al. (2023) and MapReduce LoRA Chen et al. (2026) train a separate model or LoRA expert for each reward and merge their weights afterwards. Closest to our work is MARBLE Zhao et al. (2026), which keeps a single model, combines the normalized reward gradients of DiffusionNFT with MGDA, and averages the resulting weights over time with an EMA. We keep the single-model setting of MARBLE but smooth the reward gradients themselves before the weights are computed, and add a preference anchor to the weights.

Gradient-based multi-objective optimization. When several objectives are optimized together, their gradients can conflict, and many methods decide the shared update from the individual gradients instead of a fixed weighting. MGDA Désidéri (2012) takes the minimum-norm convex combination of the objective gradients as a common descent direction, and was brought to deep multi-task learning by Sener & Koltun (2018) and Lin et al. (2019). Many later methods change how the gradients are combined, for example by normalizing their magnitudes Chen et al. (2018), removing conflicting components Yu et al. (2020), or equalizing the improvement across tasks Liu et al. (2021); Navon et al. (2022); Liu et al. (2023). Our preference anchor is borrowed from SDMGrad Xiao et al. (2023), which pulls the MGDA direction toward a weighted combination of the gradients. A separate line of work looks at what happens when the gradients are noisy. Liu & Vicente (2024) show that the MGDA direction is biased even when each gradient estimate is unbiased, and MoCo Fernando et al. (2023) corrects this bias with a provably convergent stochastic update; MoDo Chen et al. (2023b) further analyzes the trade-off between optimization and generalization in this setting. In diffusion RL, the gradients are noisy because they are estimated from images sampled from the policy itself, and we study how this noise affects both the combined direction and Pareto stationarity.

3 Methodology

We propose *Smooth Then Balance* (STB), which first smooths the reward-specific gradients (Section 3.2) and then balances them with a preference anchor (Section 3.3). Fig. 2 gives an overview, and Section 3.4 presents the theoretical analysis.

3.1 Preliminaries

Diffusion RL objective. We adopt DiffusionNFT Zheng et al. (2026) as the underlying RL training objective. Given a prompt c, the diffusion policy generates a sample x_0, whose reward is converted into an optimality probability $o(x_0, c) \in [0, 1]$. For a noisy latent x_t at diffusion timestep t, let v denote the target velocity, $v_\theta(x_t, c, t)$ the prediction of the current policy, and $v^{\text{old}}(x_t, c, t)$ the prediction of the data-collection policy. DiffusionNFT trains the policy with the per-sample loss

$$\ell_\theta(o) = o \left\| v_\theta^+ - v \right\|_2^2 + (1 - o) \left\| v_\theta^- - v \right\|_2^2, \qquad v_\theta^\pm = v^{\text{old}} \pm \beta \left(v_\theta - v^{\text{old}} \right), \tag{1}$$

where $v_\theta^\pm$ are the implicit positive and negative policies, $\beta > 0$ controls the reinforcement strength, and the dependence on (x_t, c, t) is omitted for brevity.

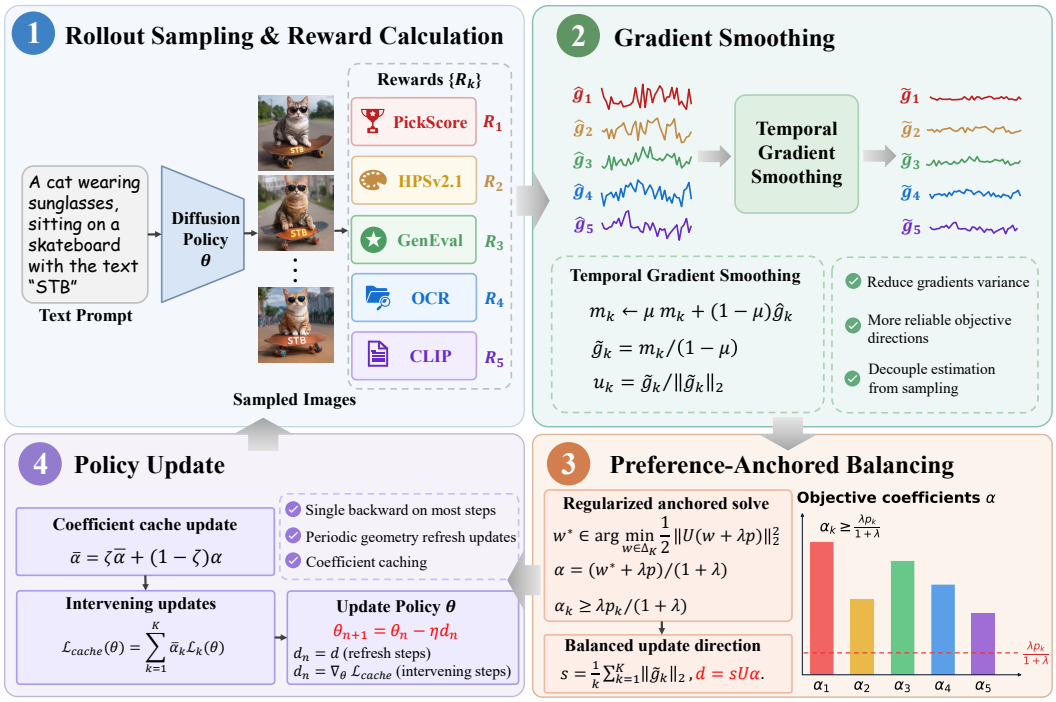


Figure 2: **Overview of STB. (1)** The diffusion policy samples images for each prompt, which are scored by K rewards. **(2)** Gradient smoothing maintains a separate bias-corrected EMA of each objective gradient and normalizes it to a unit direction u_k. **(3)** Preference-anchored balancing solves for the weights α over the smoothed directions. **(4)** The policy is updated with the combined gradient d; between rounds, the cached coefficients $\bar{\alpha}$ weight the reward losses.

Multi-objective formulation. For K rewards $R_1, \ldots, R_K$, let $\mathcal{L}_k(\theta) = \mathbb{E}[\ell_\theta(o_k)]$ be the expected DiffusionNFT loss of reward k, where o_k is the optimality probability induced by R_k and the expectation is over prompts, sampled images, diffusion timesteps, and forward-process noise. We jointly minimize the K objectives without fixed scalar weights,

$$\min_\theta \ \mathcal{L}(\theta) = \big(\mathcal{L}_1(\theta), \ldots, \mathcal{L}_K(\theta)\big), \tag{2}$$

in the Pareto sense: θ is Pareto optimal if no other parameter improves one objective without worsening another, and a necessary condition is Pareto stationarity, where some convex combination of the objective gradients vanishes Désidéri (2012). In practice, each objective is estimated on a sampled training batch $\mathcal{B}$ as

$$\widehat{\mathcal{L}}_k(\theta) = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \ell_\theta(o_{ik}), \tag{3}$$

where o_{ik} is the optimality probability induced by reward R_k for sample i. Gradient-based methods such as MGDA Désidéri (2012) update the policy along a weighted combination of the objective gradients, and STB improves both the gradients (Section 3.2) and the weights (Section 3.3).

3.2 Smooth: Estimate Objective Gradients Before Computing the Weights

Let $\widehat{g}_{k,r} = \nabla_\theta \widehat{\mathcal{L}}_k(\theta_r)$ be the gradient of Eq. (3) for reward k at training round r and $\widehat{G}_r = [\widehat{g}_{1,r}, \ldots, \widehat{g}_{K,r}]$ the gradient matrix. Since these gradients are estimated from a single batch, they vary substantially across rounds. STB therefore maintains an exponential moving average (EMA) of the gradient matrix,

$$E_r = \mu E_{r-1} + (1-\mu)\widehat{G}_r, \qquad \widetilde{G}_r = \frac{E_r}{1-\mu^r}, \tag{4}$$

where $E_0 = 0$, $\mu \in [0, 1)$ controls the smoothing strength, and the factor $1 - \mu^r$ corrects the initialization bias. The k-th column $\widetilde{g}_{k,r}$ of $\widetilde{G}_r$ is the smoothed gradient of reward k. We then

Algorithm 1 One round of STB

Require: Objective gradients $\{\widehat{g}_{k,r}\}_{k=1}^{K}$, EMA state E_{r-1}, coefficient cache $\bar{\alpha}_{r-1}$, smoothing coefficient μ, anchor strength λ, cache strength ζ, and preference weights p

1: Update E_r and compute the smoothed gradient matrix $\widetilde{G}_r$ using Eq. (4)
2: Normalize the columns of $\widetilde{G}_r$ to obtain U_r using Eq. (5)
3: Solve Eq. (7) to obtain the adaptive weights $w_r^\star$
4: Combine $w_r^\star$ with p using Eq. (8) to obtain α_r
5: Form the combined gradient d_r using Eq. (9)
6: Update the coefficient cache $\bar{\alpha}_r$ using Eq. (14)
7: **return** d_r for the current update and $\bar{\alpha}_r$ for the next $H-1$ updates (Eq. (15))

normalize the columns,

$$U_r = \widetilde{G}_r \operatorname{diag}\big(\|\widetilde{g}_{1,r}\|_2, \ldots, \|\widetilde{g}_{K,r}\|_2\big)^{-1}, \tag{5}$$

which is used in place of $\widehat{G}_r$ to compute the weights in Section 3.3.

3.3 Balance: Anchor the Allocation to a Preference

For a single round, we omit the round index and write $U = U_r$ from Eq. (5). Given U, MGDA computes the weights by solving the quadratic program

$$w^{\mathrm{MGDA}} \in \arg\min_{w \in \Delta_K} \frac{1}{2}\big\|Uw\big\|_2^2, \tag{6}$$

where $\Delta_K = \{w \in \mathbb{R}^K_{\geq 0} : \sum_k w_k = 1\}$ is the probability simplex, and updates the policy along the common descent direction Uw^{MGDA}. Because these weights are determined solely by the gradient directions, MGDA can assign very small weights to some rewards, and smoothing the gradients does not change this. STB therefore replaces Eq. (6) with an anchored problem adapted from SDMGrad Xiao et al. (2023), which provides a minimum weight for each reward.

Let $p \in \Delta_K$ be a fixed preference vector with $p_k > 0$, and we use the uniform preference $p_k = 1/K$. STB computes the weights by solving

$$w^\star \in \arg\min_{w \in \Delta_K} \frac{1}{2}\left\|U(w + \lambda p)\right\|_2^2, \tag{7}$$

where $\lambda \geq 0$ controls the strength of the preference anchor, and combines the rewards with the weights

$$\alpha = \frac{w^\star + \lambda p}{1 + \lambda}. \tag{8}$$

Since $w^\star$ is nonnegative, every reward receives a weight of at least $\lambda p_k/(1+\lambda)$ (the dashed line in Fig. 2, panel 3), while the remaining weight $1/(1+\lambda)$ is still determined by the gradient directions. Setting $\lambda = 0$ recovers MGDA in Eq. (6).

Finally, because U contains normalized directions, STB restores the gradient magnitude with the average norm of the smoothed gradients and forms the reward gradient for the policy update,

$$s = \frac{1}{K}\sum_{k=1}^{K} \|\widetilde{g}_k\|_2, \qquad d = sU\alpha, \tag{9}$$

where $U\alpha = \sum_k \alpha_k u_k$ is the combined direction. Algorithm 1 summarizes one round. For efficiency, we adopt the amortization and coefficient EMA of MARBLE Zhao et al. (2026), where the weights are recomputed every H optimizer updates, and on the intervening updates the reward losses are weighted by an EMA of the coefficients α (Appendix A).

3.4 Theoretical Analysis

We analyze the direction d_r of Algorithm 1, treating the reward objectives as fixed differentiable functions $f_1, \ldots, f_K$. Let $g_{k,r} = \nabla f_k(\theta_r)$ be the true gradient and $\varepsilon_r = \max_k \|\widetilde{g}_{k,r} - g_{k,r}\|_2$ the largest estimation error of the nonzero smoothed gradients.

The weights α of STB range over the anchored simplex

$$\mathcal{A}_{\lambda,p} = \Big\{\alpha \in \Delta_K : \alpha_k \geq \frac{\lambda p_k}{1+\lambda}\Big\}, \tag{10}$$

and since every $\alpha \in \mathcal{A}_{\lambda,p}$ corresponds to $w = (1+\lambda)\alpha - \lambda p \in \Delta_K$, Eq. (7) is equivalent to minimizing $\|U\alpha\|_2$ over $\mathcal{A}_{\lambda,p}$. For any matrix V with unit-norm columns, let $q(V)$ be the minimum-norm vector in $V\mathcal{A}_{\lambda,p} = \{V\alpha : \alpha \in \mathcal{A}_{\lambda,p}\}$. The combined direction of STB is then exactly $U_r\alpha_r = q(U_r)$, which is unique even when α_r is not. The first proposition compares it with the direction that STB would produce from the true gradients.

Proposition 3.1 (Stability of the combined direction)**.** *Let* $U_{\star,r} = [g_{1,r}/\|g_{1,r}\|_2, \ldots, g_{K,r}/\|g_{K,r}\|_2]$*, and assume* $\min_k \|g_{k,r}\|_2 \geq \gamma > 0$*. Then*

$$\|q(U_r) - q(U_{\star,r})\|_2^2 \leq \frac{4\varepsilon_r}{\gamma}. \tag{11}$$

The proof is given in Appendix B.2.

Remark 3.1 (Why smooth the gradients). The bound tightens as the gradient estimation error ε_r decreases and does not depend on the weights α_r, so it is reducing ε_r, rather than stabilizing the weights, that makes the combined direction more accurate. For a fixed gradient mean and independent zero-mean noise, the EMA in Eq. (4) asymptotically reduces the noise variance by a factor of $(1-\mu)/(1+\mu)$. Because the EMA also introduces a tracking lag as the gradients change during training, smoothing lowers ε_r when this noise reduction outweighs the lag (Appendix B.1).

The next proposition relates d_r to Pareto stationarity without requiring the lower bound γ.

Proposition 3.2 (Pareto-stationarity certificate)**.** *Define* $G_r = [g_{1,r}, \ldots, g_{K,r}]$ *and the Pareto-stationarity residual*

$$\mathcal{P}(\theta_r) = \min_{a \in \Delta_K} \|G_r a\|_2, \tag{12}$$

where $\mathcal{P}(\theta_r) = 0$ *is first-order Pareto stationarity. For any allocation* $\alpha_r \in \mathcal{A}_{\lambda,p}$*, including an inexact solution of Eq. (7), the STB direction* $d_r = s_r U_r \alpha_r$ *satisfies*

$$\mathcal{P}(\theta_r) \leq \underbrace{K\|d_r\|_2}_{\textit{update norm}} + \underbrace{\varepsilon_r}_{\textit{estimation error}}. \tag{13}$$

Consequently, $\|d_r\|_2 \to 0$ *and* $\varepsilon_r \to 0$ *imply* $\mathcal{P}(\theta_r) \to 0$*. The proof is given in Appendix B.3.*

Remark 3.2 (Role of the preference anchor). The bound is a sufficient condition. If both $\|d_r\|_2$ and ε_r are small, the Pareto-stationarity residual is certified to be small, and smoothing contributes by reducing ε_r. A large ε_r only makes the bound loose and does not imply a large residual. Since the bound holds for every $\alpha_r \in \mathcal{A}_{\lambda,p}$, the certificate remains valid after adding the anchor. The anchor, however, only lower-bounds the weights, and it guarantees neither that every reward improves nor that the update norm does not increase.

4 Experiments

4.1 Experimental Setup

Training and implementation. We fine-tune SD3.5-Medium (Esser et al., 2024) with LoRA on a 1:1:1 mixture of Pick-a-Pic (Kirstain et al., 2023), GenEval (Ghosh et al., 2023), and OCR (Liu et al., 2026) training prompts, jointly optimizing five rewards, namely PickScore (Kirstain et al., 2023), HPSv2.1 (Wu et al., 2023a), CLIPScore (Hessel et al., 2021), GenEval (Ghosh et al., 2023), and OCR accuracy Cui et al. (2025). STB uses $\mu = 0.9$, $\lambda = 0.3$, and uniform p, and the remaining hyperparameters follow the default setting of MARBLE (Zhao et al., 2026). Data statistics and the full list of hyperparameters are given in Appendix A.

Evaluation. Following DiffusionNFT (Zheng et al., 2026), we evaluate GenEval (Ghosh et al., 2023) and OCR accuracy Cui et al. (2025) on their test prompts and all model-based metrics on DrawBench (Saharia et al., 2022). Besides the three model-based training rewards, PickScore Kirstain et al. (2023), CLIPScore Hessel et al. (2021), and HPSv2.1 Wu et al. (2023a),

Table 1: Comparison across eight evaluation metrics (higher is better). All rows are evaluated with our pipeline. †: released checkpoints; §: reproduced training. **Bold** and underline denote the best and second-best scores among fine-tuned SD3.5-M. Green cells mark training rewards.

Model	Rule-Based		Model-Based					
	GenEval	OCR	PickScore	CLIPScore	HPSv2.1	Aesthetic	ImgRwd	UniRwd
SD-XL Podell et al. (2024)	0.55	0.14	22.42	0.287	0.280	5.60	0.76	2.93
SD3.5-L Esser et al. (2024)	0.71	0.68	22.91	0.289	0.288	5.50	0.96	3.25
FLUX.1-Dev Labs (2024)	0.66	0.59	22.84	0.295	0.274	5.71	0.96	3.27
SD3.5-M Esser et al. (2024)	0.63	0.59	22.34	0.285	0.279	5.36	0.85	3.03
+ MapReduce Lora† Chen et al. (2026)	0.89	0.91	22.81	0.290	0.296	5.47	1.23	3.14
+ DiffusionNFT† Zheng et al. (2026)	<u>0.94</u>	0.89	**23.80**	<u>0.293</u>	<u>0.329</u>	**6.00**	**1.49**	**3.35**
+ Flow-GRPO§ Liu et al. (2026)	0.85	0.87	22.59	0.286	0.309	5.53	1.40	3.15
+ MGDA§ Zhao et al. (2026)	0.93	<u>0.95</u>	22.73	0.283	0.305	5.65	1.28	3.14
+ STB (ours)	**0.96**	**0.96**	<u>22.91</u>	**0.297**	**0.330**	<u>5.70</u>	<u>1.46</u>	<u>3.29</u>

we report three held-out metrics that are never optimized, namely Aesthetic score Schuhmann et al. (2022), ImageReward (Xu et al., 2023a), and UniReward (Wang et al., 2025).

Baselines. We compare with pre-trained models, namely SD-XL (Podell et al., 2024), SD3.5-Large (Esser et al., 2024), FLUX.1-Dev (Labs, 2024), and SD3.5-Medium (Esser et al., 2024), and with multi-objective diffusion RL methods fine-tuned from SD3.5-Medium. MapReduce LoRA (Chen et al., 2026) and DiffusionNFT (Zheng et al., 2026) are evaluated from their released multi-reward checkpoints, Flow-GRPO (Liu et al., 2026) is our reproduction with a weighted sum of the five rewards, and MGDA is our reproduction following MARBLE (Zhao et al., 2026), with the same setting as STB. All models are evaluated with the same pipeline.

4.2 Results and Analysis

Main results. Among the fine-tuned models in Table 1, STB achieves the highest GenEval, OCR, CLIPScore, and HPSv2.1 scores and the second-highest scores on the remaining four metrics. Compared with MGDA, which is trained with the same data, rewards, and configuration, STB improves all eight metrics. The largest gains appear on HPSv2.1 (0.305 to 0.330), CLIPScore (0.283 to 0.297), and ImageReward (1.28 to 1.46), and PickScore, which receives little weight under MGDA, also improves from 22.73 to 22.91. The improvements extend to the three held-out metrics that are never optimized, indicating that the gains are not limited to the training rewards. STB also outperforms MapReduce LoRA and Flow-GRPO on all eight metrics. Compared with the released DiffusionNFT checkpoint, STB is clearly stronger on the rule-based metrics, with GenEval 0.96 versus 0.94 and OCR 0.96 versus 0.89, while DiffusionNFT remains stronger on PickScore and the held-out metrics. Notably, STB built on SD3.5-M matches or exceeds the larger SD3.5-L on all eight metrics and outperforms FLUX.1-Dev on seven of them. Since MGDA and STB share the same data, rewards, and configuration, the gains over MGDA can be attributed to the two components of STB, which we examine next.

Ablation of STB components. Table 2 shows that each component improves over MGDA on its own. Adding smoothing alone improves all eight metrics, for example raising OCR from 0.95 to 0.97, and adding the anchor alone improves seven metrics while keeping OCR unchanged, for example raising HPSv2.1 from 0.305 to 0.319. The two components improve different metrics. Smoothing brings larger gains on OCR and UniReward (3.14 to 3.25), whereas the anchor brings larger gains on the preference-related metrics, raising PickScore from 22.73 to 22.81, HPSv2.1 from 0.305 to 0.319, and ImageReward from 1.28 to 1.42, consistent with its role of preventing these rewards from being underweighted. Combining the two achieves the best results on seven of the eight metrics, with OCR slightly below the smoothing-only variant (0.96 *vs.* 0.97), and the combined gains on HPSv2.1 (0.330) and ImageReward (1.46) exceed those of either component alone, indicating that the two components are complementary.

Sensitivity to the smoothing strength. Fig. 3 compares $\mu \in \{0.5, 0.7, 0.9\}$ with $\lambda = 0.3$. During roughly the first 600 updates, $\mu = 0.9$ attains the highest PickScore, HPSv2.1, and mean normalized reward, while $\mu = 0.5$ shows larger drops. The three settings later approach each other, but $\mu = 0.5$ still ends lowest on PickScore and HPSv2.1 and $\mu = 0.9$ ends slightly lower on CLIPScore, so

Table 2: Ablation of the two components of STB (higher is better). The first row, with neither component, is MGDA and the last row is STB. **Bold** marks the best score in each column.

Smooth	Anchor	GenEval	OCR	PickScore	CLIPScore	HPSv2.1	Aesthetic	ImgRwd	UniRwd
×	×	0.93	0.95	22.73	0.283	0.305	5.65	1.28	3.14
✓	×	0.94	**0.97**	22.74	0.288	0.310	5.68	1.33	3.25
×	✓	0.94	0.95	22.81	0.289	0.319	5.67	1.42	3.21
✓	✓	**0.96**	0.96	**22.91**	**0.297**	**0.330**	**5.70**	**1.46**	**3.29**

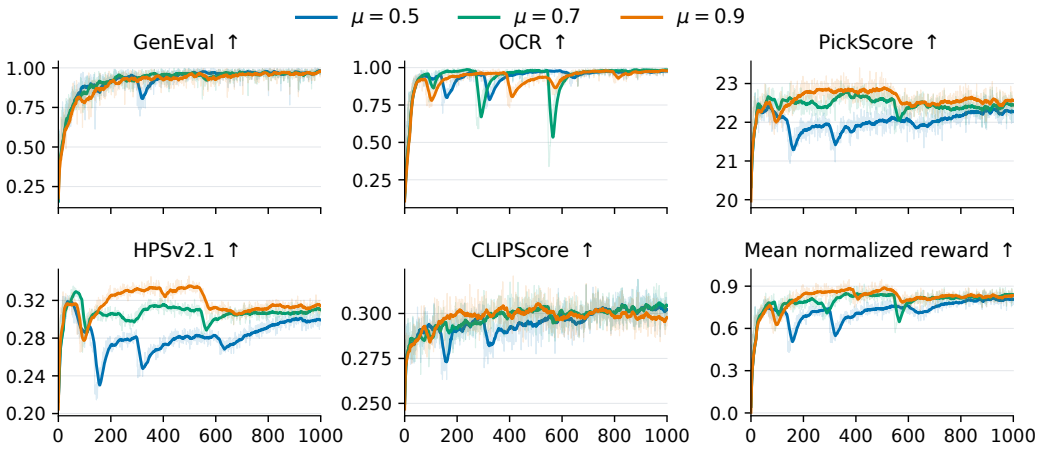


Figure 3: **Training trajectories at different momentum strengths.** Horizontal axes show optimizer steps. The final panel averages five rewards after per-reward min–max normalization.

stronger smoothing is not uniformly better on every reward. This behavior matches the trade-off in Remark 3.1: a larger μ suppresses more gradient noise, which helps most early in training, but also increases the lag behind the current gradient. We use $\mu = 0.9$ by default, as it performs best during most of training.

Gradient stability. Fig. 4 compares the reward gradients with and without smoothing during STB training, using two measures computed between consecutive rounds: the angle between the gradients of the same reward, and the relative change of the normalized Gram matrix $U_r^\top U_r$, which contains the pairwise cosine similarities that determine the weights in Eq. (7). Without smoothing, the median direction change of each reward gradient is $79.2°$–$89.6°$, so consecutive estimates of the same gradient are nearly orthogonal and the weights are computed from directions that are dominated by sampling noise. Smoothing reduces this change to $10.2°$–$23.3°$ for all five rewards and reduces the median relative change of the Gram matrix from 31.5% to 6.25%. Both the individual gradient directions and their pairwise relations thus vary much less between rounds. This stability carries over to the weights and to the combined direction, which is consistent with Proposition 3.1.

Qualitative comparison. Fig. 5 compares object attributes, text rendering, and spatial composition across the four methods. For attribute binding, STB is the only method that renders a fully blue pizza next to the yellow glove, whereas SD3.5-M and MapReduce LoRA color the plate or tray instead and DiffusionNFT only partially recolors the toppings. For text rendering, SD3.5-M misspells both "Street Art Rules" and "Google Research Pizza Cafe", and DiffusionNFT repeats "Cafe" in the latter. MapReduce LoRA spells the text correctly but renders it small, while STB renders all requested text legibly and prominently, including the engraved "Always Forever" that is missing from SD3.5-M and only partially legible in DiffusionNFT. For the remaining color and spatial prompts, STB preserves both requested colors and places the tie above the sink, whereas SD3.5-M hangs the tie below the sink. Overall, STB satisfies all requested constraints in every example, while each baseline fails on at least one of them. More qualitative results are provided in Appendix D.

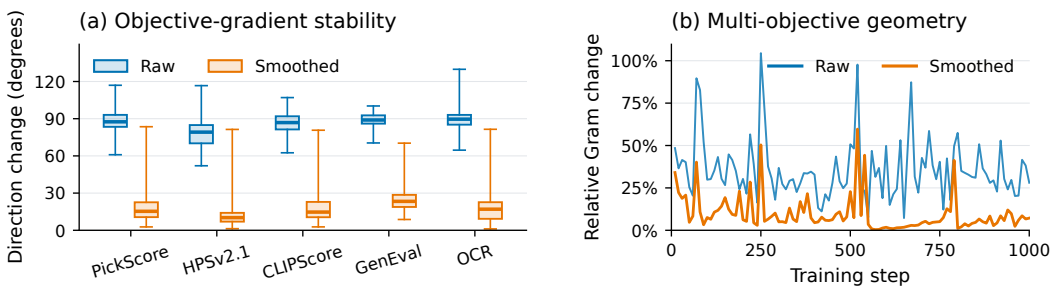


Figure 4: **Gradient stability.** Left: inter-refresh direction changes for five objectives. Right: relative change of the normalized Gram matrix. Raw and smoothed gradients are paired along one STB trajectory; lower values indicate greater stability.

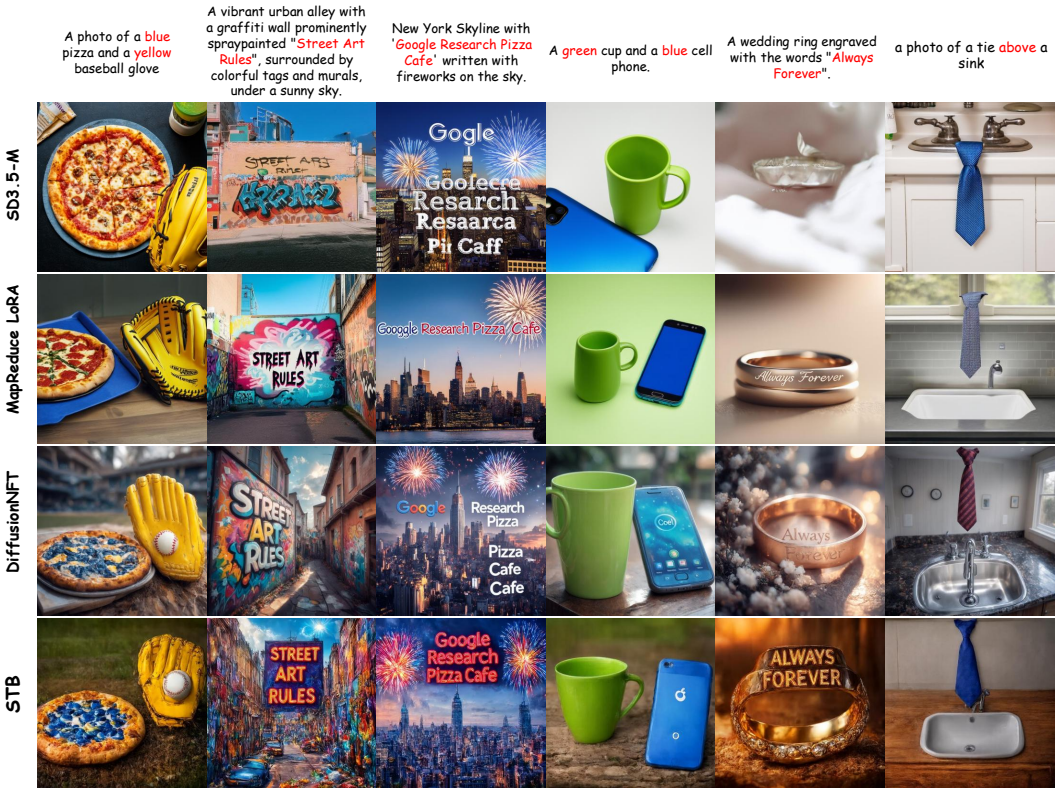


Figure 5: **Qualitative comparison.** Each column compares SD3.5-M, MapReduce LoRA, DiffusionNFT, and STB for the prompt shown above. Red text highlights key properties.

5 Conclusion

We studied MGDA for multi-objective diffusion RL and identified two problems in this setting. The reward gradients are noisy because they are estimated from sampled images, and the resulting weights can be highly imbalanced, leaving some rewards under-optimized. STB addresses the first problem by smoothing each reward gradient before the weights are computed and the second by balancing the smoothed gradients with a preference anchor. Our theoretical results show that the error of the STB direction is bounded by the gradient estimation error, and that a small STB update guarantees closeness to Pareto stationarity once this error is small. On five-reward fine-tuning of SD3.5-Medium, STB outperforms MGDA on all eight evaluation metrics.

AI USE STATEMENT

We used large language models (LLMs) to polish the writing of this paper and to assist with code implementation, such as building the data pipeline and plotting figures. The core algorithm design was carried out by the authors. All LLM-assisted text and code were reviewed and verified by the authors, who take full responsibility for the content of this paper.

REFERENCES

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pp. 4965–4987, 2024.

Chieh-Yun Chen, Zhonghao Wang, Qi Chen, Zhifan Ye, Min Shi, Yue Zhao, Yinan Zhao, Hui Qu, Wei-An Lin, Yiru Shen, et al. Mapreduce lora: Advancing the pareto front in multi-preference optimization for generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 34375–34384, 2026.

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, 2023a.

Lisha Chen, Heshan Fernando, Yiming Ying, and Tianyi Chen. Three-way trade-off in multi-objective learning: Optimization, generalization and conflict-avoidance. *Advances in Neural Information Processing Systems*, 36:70045–70093, 2023b.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International conference on machine learning*, pp. 794–803. PMLR, 2018.

Kevin Clark, Paul Vicol, Kevin Swersky, and David Fleet. Directly fine-tuning diffusion models on differentiable rewards. In *International Conference on Learning Representations*, volume 2024, pp. 4793–4822, 2024.

Cheng Cui, Ting Sun, Manhui Lin, Tingquan Gao, Yubo Zhang, Jiaxuan Liu, Xueqing Wang, Zelun Zhang, Changda Zhou, Hongen Liu, Yue Zhang, Wenyu Lv, Kui Huang, Yichao Zhang, Jing Zhang, Jun Zhang, Yi Liu, Dianhai Yu, and Yanjun Ma. Paddleocr 3.0 technical report, 2025. URL https://arxiv.org/abs/2507.05595.

Jean-Antoine Désidéri. Multiple-gradient descent algorithm (mgda) for multiobjective optimization. *Comptes Rendus. Mathématique*, 350(5-6):313–318, 2012.

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 *International Conference on Machine Learning*, 2024.

Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in neural information processing systems*, 36:79858–79885, 2023.

Heshan Devaka Fernando, Han Shen, Miao Liu, Subhajit Chaudhury, Keerthiram Murugesan, and Tianyi Chen. Mitigating gradient bias in multi-objective learning: A provably convergent approach. In *The eleventh international conference on learning representations*, 2023.

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.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 7514–7528, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

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.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in Neural Information Processing Systems*, 34:21696–21707, 2021.

Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. Pick-a-pic: An open dataset of user preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:36652–36663, 2023.

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

Seung Hyun Lee, Yinxiao Li, Junjie Ke, Innfarn Yoo, Han Zhang, Jiahui Yu, Qifei Wang, Fei Deng, Glenn Entis, Junfeng He, et al. Parrot: Pareto-optimal multi-reward reinforcement learning framework for text-to-image generation. In *European Conference on Computer Vision*, pp. 462–478. Springer, 2024.

Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qing-Fu Zhang, and Sam Kwong. Pareto multi-task learning. *Advances in Neural Information Processing Systems*, 32, 2019.

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

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34:18878–18890, 2021.

Bo Liu, Yihao Feng, Peter Stone, and Qiang Liu. Famo: Fast adaptive multitask optimization. *Advances in Neural Information Processing Systems*, 36:57226–57243, 2023.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online rl. *Advances in Neural Information Processing Systems*, 38:40783–40818, 2026.

Suyun Liu and Luis Nunes Vicente. The stochastic multi-gradient algorithm for multi-objective optimization and its application to supervised machine learning. *Annals of Operations Research*, 339(3):1119–1148, 2024.

Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Multi-task learning as a bargaining game. *arXiv preprint arXiv:2202.01017*, 2022.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, volume 139, pp. 8162–8171, 2021.

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. In *International Conference on Learning Representations*, volume 2024, pp. 1862–1874, 2024.

Alexandre Rame, Guillaume Couairon, Corentin Dancette, Jean-Baptiste Gaya, Mustafa Shukor, Laure Soulier, and Matthieu Cord. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. *Advances in Neural Information Processing Systems*, 36:71095–71134, 2023.

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.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in Neural Information Processing Systems*, 31, 2018.

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 *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8228–8238. IEEE, 2024.

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.

Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. *arXiv preprint arXiv:2306.09341*, 2023a.

Xiaoshi Wu, Keqiang Sun, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score: Better aligning text-to-image models with human preference. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2096–2105. IEEE, 2023b.

Peiyao Xiao, Hao Ban, and Kaiyi Ji. Direction-oriented multi-objective learning: Simple and provable stochastic algorithms. *Advances in Neural Information Processing Systems*, 36:4509–4533, 2023.

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

Yilun Xu, Shangyuan Tong, and Tommi S. Jaakkola. Stable target field for reduced variance score estimation in diffusion models. In *International Conference on Learning Representations*, 2023b.

Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, and Ping Luo. DanceGRPO: Unleashing GRPO on visual generation. *arXiv preprint arXiv:2505.07818*, 2025.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33:5824–5836, 2020.

Canyu Zhao, Hao Chen, Yunze Tong, Yu Qiao, Jiacheng Li, and Chunhua Shen. Marble: Multi-aspect reward balance for diffusion rl. *arXiv preprint arXiv:2605.06507*, 2026.

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. Diffusionnft: Online diffusion reinforcement with forward process. In *International Conference on Learning Representations*, volume 2026, pp. 134129–134150, 2026.

Appendix

A Implementation Details

Training data and evaluation. The training prompts are drawn in a 1:1:1 ratio from Pick-a-Pic (Kirstain et al., 2023) (25,432 prompts) and the GenEval (50,000 prompts) and OCR (19,653 prompts) training sets of Flow-GRPO (Liu et al., 2026), following DiffusionNFT (Zheng et al., 2026). PickScore, HPSv2.1, and CLIPScore are computed on all prompts, whereas GenEval and OCR accuracy are computed only on prompts from their own source. We train LoRA adapters (rank 32, $\alpha = 64$) at 512×512 resolution with 40-step rollouts and 24 images per prompt, giving 1,152 rollout images per optimizer update. At evaluation, all methods are sampled with 40 steps and without classifier-free guidance. All training runs use 16 NVIDIA GH200 GPUs with 96 GB of memory each.

Amortized update. We follow MARBLE's amortization and coefficient EMA (Zhao et al., 2026); these are inherited implementation choices, not STB components. Computing K objective gradients and solving Eq. (7) at every optimizer update would require K backward passes per microbatch, so Algorithm 1 is applied only periodically. Let n index optimizer updates and H be the interval between rounds. Round r takes place at update $n_r = (r-1)H$, so the main-text parameter θ_r is the policy at that update. At each round, we accumulate and globally synchronize the K objective gradients over all microbatches before applying Algorithm 1, and cache the effective coefficients with an EMA of strength $\zeta \in [0, 1)$:

$$\bar{\alpha}_r = \zeta \bar{\alpha}_{r-1} + (1-\zeta)\alpha_r, \qquad \bar{\alpha}_0 = p. \tag{14}$$

At the intervening updates $n_r < n < n_{r+1}$, we optimize the scalarized loss

$$\mathcal{L}_{\mathrm{cache},n}(\theta) = \sum_{k=1}^{K} \bar{\alpha}_{k,r} \widehat{\mathcal{L}}_{k,n}(\theta), \tag{15}$$

where $\widehat{\mathcal{L}}_{k,n}$ is the empirical objective $\widehat{\mathcal{L}}_k$ of Section 3.1 on the batch at update n; this requires one backward pass per microbatch. The reward gradient passed to the optimizer is therefore

$$d_n = \begin{cases} d_r, & n = n_r, \\ \nabla_\theta \mathcal{L}_{\mathrm{cache},n}(\theta_n), & n_r < n < n_{r+1}. \end{cases} \tag{16}$$

The coefficient cache weights the current losses, rather than replaying the smoothed, normalized direction from the preceding round. Both branches add the gradient of the shared KL term and use the same optimizer (AdamW).

Configuration. Table 3 lists the training hyperparameters shared by STB and our reproduced MGDA baseline. The numerical implementation excludes nonfinite or negligible-norm directions, renormalizes the preference over retained directions, and skips the solve when none remain; the main-text analysis assumes all K directions are usable.

B Theoretical Analysis of STB

This appendix proves the results of Section 3.4. Appendix B.1 bounds the estimation error ε_r of the smoothed gradients, Appendix B.2 proves Proposition 3.1, and Appendix B.3 proves Proposition 3.2. Appendix B.4 states the scope of these results.

Scope and notation. We consider fixed differentiable losses $f_k : \mathbb{R}^D \to \mathbb{R}$, not a changing sequence of detached NFT surrogates. As in Section 3, r indexes rounds and θ_r is the parameter at round r. We write $g_{k,r} = \nabla f_k(\theta_r)$, $G_r = [g_{1,r}, \ldots, g_{K,r}]$, $\widetilde{G}_r = [\widetilde{g}_{1,r}, \ldots, \widetilde{g}_{K,r}]$, $h_{k,r} = \|\widetilde{g}_{k,r}\|_2 > 0$, $U_r = \widetilde{G}_r \operatorname{diag}(h_r)^{-1}$, and $s_r = K^{-1} \sum_k h_{k,r}$, so that $d_r = s_r U_r \alpha_r$ as in Section 3.3. The feasible set $\mathcal{A}_{\lambda,p}$ is the anchored simplex in Eq. (10), and $q(V)$ is the minimum-norm element of $V\mathcal{A}_{\lambda,p}$. Our analysis uses the convex-hull stationarity principle underlying MGDA (Désidéri, 2012), the anchored subproblem of SDMGrad (Xiao et al., 2023), and

Table 3: Training hyperparameters.

	Hyperparameter	Value
Model	Base model	SD3.5-Medium
	LoRA rank / α	32 / 64
	Resolution	512×512
	Mixed precision	bf16
Rollout	Prompt sources (mixing ratio)	Pick-a-Pic, GenEval, OCR (1:1:1)
	Prompts / images per prompt per update	48 / 24
	Rollout images per update	1,152
	Sampler / steps	deterministic DPM2 / 40
	Classifier-free guidance scale	1.0
DiffusionNFT	NFT strength β	0.1
	KL weight	10^{-4}
	Advantage clipping $A_{\max}$	5
	Training timestep fraction	0.99
Optimizer	Optimizer	AdamW
	Learning rate	3×10^{-4}
	(β_1, β_2) / ϵ / weight decay	$(0.9, 0.999)$ / 10^{-8} / 10^{-4}
	Gradient clipping (max norm)	1.0
STB	Gradient smoothing μ	0.9
	Anchor strength λ	0.3
	Preference p	uniform ($p_k = 0.2$)
	Weight-update interval H	10
	Coefficient EMA ζ	0.7
Compute	GPUs	16× NVIDIA GH200 (96 GB)

the distinction between gradient-tracking error and direction-selection error emphasized by stochastic MOO analyses (Fernando et al., 2023); the results below are derived for the stated updates and are not a direct application of those algorithms' theorems.

B.1 EMA TRACKING ERROR: NOISE REDUCTION VERSUS LAG

Assume $\widehat{g}_{k,r} = g_{k,r} + \xi_{k,r}$, where θ_r is measurable before drawing $\xi_{k,r}$, $\mathbb{E}[\xi_{k,r} \mid \mathcal{F}_{r-1}] = 0$, and $\mathbb{E}[\|\xi_{k,r}\|_2^2 \mid \mathcal{F}_{r-1}] \le \sigma_k^2$. Correlation between different objectives at the same round is allowed. Let each ∇f_k be L-Lipschitz. The bias-corrected EMA in Eq. (4) has deterministic weights

$$b_{r,j} = \frac{(1-\mu)\mu^{r-j}}{1-\mu^r}, \qquad 1 \le j \le r, \tag{17}$$

which sum to one. This gives the exact decomposition

$$\widetilde{g}_{k,r} - g_{k,r} = \underbrace{\sum_{j=1}^{r} b_{r,j}\xi_{k,j}}_{\text{filtered noise}} + \underbrace{\sum_{j=1}^{r} b_{r,j}(g_{k,j} - g_{k,r})}_{\text{lag}}. \tag{18}$$

Proposition B.1 (Finite-round tracking bound)**.** *If $\|\theta_{j+1} - \theta_j\|_2 \le D_0$ almost surely, then*

$$\mathbb{E}\|\widetilde{g}_{k,r} - g_{k,r}\|_2^2 \le 2\sigma_k^2 \frac{1-\mu}{1+\mu}\frac{1+\mu^r}{1-\mu^r} + 2L^2 D_0^2 \left(\frac{\mu}{1-\mu}\right)^2. \tag{19}$$

Consequently, the right-hand side summed over k bounds $\mathbb{E}[\varepsilon_r^2]$.

Proof. Martingale differences at distinct rounds have zero cross moments, so the filtered-noise term has second moment at most $\sigma_k^2 \sum_j b_{r,j}^2$, and the geometric-series identity gives

$$\sum_{j=1}^{r} b_{r,j}^2 = \frac{1-\mu}{1+\mu}\frac{1+\mu^r}{1-\mu^r}. \tag{20}$$

By Lipschitz continuity, $\|g_{k,j} - g_{k,r}\|_2 \le LD_0(r-j)$, so the lag term is bounded by $LD_0 \sum_j b_{r,j}(r-j)$, where

$$\sum_{j=1}^{r} b_{r,j}(r-j) = \frac{\mu}{1-\mu} - \frac{r\mu^r}{1-\mu^r} \le \frac{\mu}{1-\mu}. \tag{21}$$

Combining both bounds with $\|a+b\|^2 \le 2\|a\|^2 + 2\|b\|^2$ proves Eq. (19); no independence between the two terms is needed. Finally, $\varepsilon_r^2 = \max_k \|\widetilde{g}_{k,r} - g_{k,r}\|_2^2 \le \sum_k \|\widetilde{g}_{k,r} - g_{k,r}\|_2^2$. □

When the gradient is fixed ($D_0 = 0$) and the noise is independent with covariance Σ_k, the lag term vanishes, the factor two is unnecessary, and Eq. (20) gives

$$\lim_{r\to\infty} \mathrm{Cov}(\widetilde{g}_{k,r}) = \frac{1-\mu}{1+\mu}\Sigma_k, \tag{22}$$

which is the variance-reduction factor stated in Section 3.2.

B.1.1 Practical implications

Proposition B.1 does not say that larger momentum is always better: increasing μ reduces the noise term but increases the lag term, whose mean age is $\mu/(1-\mu)$ rounds. With refresh interval H, D_0 measures parameter movement over H optimizer updates. If each update moves the parameters by at most Δ_θ, then $D_0 \le H\Delta_\theta$ and the lag term in Eq. (18) is at most $LH\Delta_\theta\,\mu/(1-\mu)$. It can therefore be controlled by reducing μ, refreshing more often (smaller H), or reducing the actual parameter displacement; with AdamW, the latter should be measured rather than inferred from the learning rate. At $H = 10$, $\mu = 0.9$, 0.7, and 0.5 correspond to mean ages of approximately 90, 23.3, and 10 optimizer updates, and to stationary covariance factors of approximately 0.053, 0.176, and 0.333. With constant $\mu < 1$ and non-vanishing batch noise, the bound describes an error neighborhood rather than exact tracking.

B.2 Proof of Proposition 3.1

For a matrix $V = [v_1, \dots, v_K]$ with unit-norm columns, write $q(V) = Va_V$ with $a_V \in \arg\min_{a\in\mathcal{A}_{\lambda,p}} \frac{1}{2}\|Va\|_2^2$. Since $q(V)$ is the projection of the origin onto the compact convex set $V\mathcal{A}_{\lambda,p}$, it is unique, although a_V need not be. Throughout, $U = U_r$ has columns $u_{k,r} = \widetilde{g}_{k,r}/\|\widetilde{g}_{k,r}\|_2$, $V = U_{\star,r}$ has columns $u^\star_{k,r} = g_{k,r}/\|g_{k,r}\|_2$, and $\delta_r = \max_k \|u_{k,r} - u^\star_{k,r}\|_2$.

Step 1: error of the normalized columns. For nonzero x, y,

$$\left\| \frac{x}{\|x\|_2} - \frac{y}{\|y\|_2} \right\|_2 = \left\| \frac{x-y}{\|y\|_2} + x\Big(\frac{1}{\|x\|_2} - \frac{1}{\|y\|_2}\Big) \right\|_2 \le \frac{\|x-y\|_2}{\|y\|_2} + \frac{\big|\|y\|_2 - \|x\|_2\big|}{\|y\|_2} \le \frac{2\|x-y\|_2}{\|y\|_2}. \tag{23}$$

Taking $x = \widetilde{g}_{k,r}$ and $y = g_{k,r}$, and using $\|\widetilde{g}_{k,r} - g_{k,r}\|_2 \le \varepsilon_r$ and $\|g_{k,r}\|_2 \ge \gamma$,

$$\delta_r = \max_k \|u_{k,r} - u^\star_{k,r}\|_2 \le \frac{2\varepsilon_r}{\gamma}. \tag{24}$$

Step 2: the same allocation gives nearby directions. For every $a \in \mathcal{A}_{\lambda,p} \subseteq \Delta_K$,

$$\|(U_r - U_{\star,r})a\|_2 = \Big\| \sum_k a_k(u_{k,r} - u^\star_{k,r}) \Big\|_2 \le \sum_k a_k \|u_{k,r} - u^\star_{k,r}\|_2 \le \delta_r. \tag{25}$$

Similarly, $\|q(U_r)\|_2 = \|\sum_k a_{U,k} u_{k,r}\|_2 \le \sum_k a_{U,k} = 1$, and likewise $\|q(U_{\star,r})\|_2 \le 1$.

Step 3: optimality of the two projections. Write $q = q(U_r)$ and $q^\star = q(U_{\star,r})$, with coefficients a and $a^\star$. Since q is the projection of the origin onto $U_r\mathcal{A}_{\lambda,p}$, it satisfies $\langle q, y - q\rangle \ge 0$ for all $y \in U_r\mathcal{A}_{\lambda,p}$; the same holds for $q^\star$. Choosing $y = U_r a^\star$ and $y = U_{\star,r}a$, respectively, gives

$$\langle q,\, U_r a^\star - q\rangle \ge 0, \qquad \langle q^\star,\, U_{\star,r}a - q^\star\rangle \ge 0. \tag{26}$$

Using $U_r a^\star = q^\star + (U_r - U_{\star,r})a^\star$ and $U_{\star,r}a = q + (U_{\star,r} - U_r)a$, these become

$$\begin{aligned} \langle q,\, q^\star - q\rangle + \langle q,\, (U_r - U_{\star,r})a^\star\rangle \ge 0, \\ \langle q^\star,\, q - q^\star\rangle + \langle q^\star,\, (U_{\star,r} - U_r)a\rangle \ge 0. \end{aligned} \tag{27}$$

Step 4: combine. Adding the two inequalities and using $\langle q, q^\star - q\rangle + \langle q^\star, q - q^\star\rangle = -\|q - q^\star\|_2^2$,

$$\begin{aligned}\|q - q^\star\|_2^2 &\le \langle q,\, (U_r - U_{\star,r})a^\star\rangle + \langle q^\star,\, (U_{\star,r} - U_r)a\rangle \\ &\le \|q\|_2\, \|(U_r - U_{\star,r})a^\star\|_2 + \|q^\star\|_2\, \|(U_r - U_{\star,r})a\|_2 \\ &\le 1 \cdot \delta_r + 1 \cdot \delta_r = 2\delta_r,\end{aligned} \tag{28}$$

where the second line is Cauchy–Schwarz and the third uses Step 2. Finally, Eq. (24) gives

$$\|q(U_r) - q(U_{\star,r})\|_2^2 \le 2\delta_r \le 2 \cdot \frac{2\varepsilon_r}{\gamma} = \frac{4\varepsilon_r}{\gamma}, \tag{29}$$

which is Eq. (11). □

The bound is a continuity statement: it does not claim that smoothing always reduces conflict, and it explains why coefficient oscillation alone is not evidence of an unstable aggregate direction. Near zero true gradients, unit normalization is ill-conditioned; the certificate below avoids the lower bound γ.

B.3 Proof of Proposition 3.2

Let $\alpha_r \in \mathcal{A}_{\lambda,p}$ be arbitrary (in fact $\alpha_r \in \Delta_K$ suffices), so that $d_r = s_r U_r \alpha_r$ with $s_r = K^{-1}\sum_k h_{k,r}$ and $h_{k,r} = \|\widetilde{g}_{k,r}\|_2$.

Step 1: a feasible certificate from the weights. Undo the column normalization by defining

$$Z_r = \sum_{k=1}^{K} \frac{\alpha_{k,r}}{h_{k,r}}, \qquad v_{k,r} = \frac{\alpha_{k,r}/h_{k,r}}{Z_r}. \tag{30}$$

Then $v_{k,r} \ge 0$ and $\sum_k v_{k,r} = Z_r/Z_r = 1$, so $v_r \in \Delta_K$ is a feasible point of the minimization in Eq. (12).

Step 2: the smoothed gradients combined with v_r give d_r. Using $\widetilde{g}_{k,r} = h_{k,r}u_{k,r}$,

$$\widetilde{G}_r v_r = \sum_{k=1}^{K} v_{k,r}\widetilde{g}_{k,r} = \frac{1}{Z_r}\sum_{k=1}^{K} \frac{\alpha_{k,r}}{h_{k,r}}\, h_{k,r}u_{k,r} = \frac{U_r\alpha_r}{Z_r} = \frac{d_r}{s_r Z_r}. \tag{31}$$

Step 3: replacing smoothed gradients with true gradients. Since $v_r \in \Delta_K$,

$$\|(G_r - \widetilde{G}_r)v_r\|_2 \le \sum_{k=1}^{K} v_{k,r}\|g_{k,r} - \widetilde{g}_{k,r}\|_2 \le \varepsilon_r \sum_{k=1}^{K} v_{k,r} = \varepsilon_r. \tag{32}$$

Evaluating the minimum in Eq. (12) at $a = v_r$ and applying the triangle inequality,

$$\mathcal{P}(\theta_r) \le \|G_r v_r\|_2 \le \|\widetilde{G}_r v_r\|_2 + \|(G_r - \widetilde{G}_r)v_r\|_2 \le \frac{\|d_r\|_2}{s_r Z_r} + \varepsilon_r. \tag{33}$$

Step 4: bounding the scale factor. By Cauchy–Schwarz,

$$\sum_{k=1}^{K}\sqrt{\alpha_{k,r}} = \sum_{k=1}^{K}\sqrt{h_{k,r}}\sqrt{\frac{\alpha_{k,r}}{h_{k,r}}} \le \Big(\sum_{k=1}^{K} h_{k,r}\Big)^{1/2}\Big(\sum_{k=1}^{K}\frac{\alpha_{k,r}}{h_{k,r}}\Big)^{1/2}, \tag{34}$$

and, since $\alpha_{k,r} \ge 0$ and $\sum_k \alpha_{k,r} = 1$, $\big(\sum_k \sqrt{\alpha_{k,r}}\big)^2 \ge \sum_k \alpha_{k,r} = 1$. Hence

$$s_r Z_r = \frac{1}{K}\Big(\sum_{k=1}^{K} h_{k,r}\Big)\Big(\sum_{k=1}^{K}\frac{\alpha_{k,r}}{h_{k,r}}\Big) \ge \frac{1}{K}\Big(\sum_{k=1}^{K}\sqrt{\alpha_{k,r}}\Big)^2 \ge \frac{1}{K}, \qquad \textit{i.e.} \qquad \frac{1}{s_r Z_r} \le K. \tag{35}$$

Step 5: conclusion. Substituting Eq. (35) into Eq. (33),

$$\mathcal{P}(\theta_r) \le \frac{\|d_r\|_2}{s_r Z_r} + \varepsilon_r \le K\|d_r\|_2 + \varepsilon_r, \tag{36}$$

which is Eq. (13). If $\|d_r\|_2 \to 0$ and $\varepsilon_r \to 0$, the right-hand side tends to zero, so $\mathcal{P}(\theta_r) \to 0$. □

B.4 SCOPE OF THE RESULTS

For nonconvex objectives, $\mathcal{P}(\theta_\star) = 0$ means that no direction decreases all objectives to first order; it does not exclude dominated local minima or saddles, so no global Pareto-front guarantee follows. Moreover, both propositions concern a single round with fixed objectives, whereas the complete training procedure includes cached coefficients (Eq. (15)), a KL regularizer, gradient clipping, AdamW, and changing rollout surrogates. In particular, $\sum_k \bar{\alpha}_{k,r}\widehat{g}_k$ is generally not equal to the refresh direction d_r, even if the cached coefficients stop changing. Our theoretical conclusion is therefore that accurate smoothed gradients yield a stable combined direction whose norm certifies near-Pareto stationarity, not a convergence guarantee for end-to-end diffusion training.

C TOY EXPERIMENT DETAILS

The toy optimization problem in Fig. 1 is taken from CAGrad (Liu et al., 2021), using the objective functions and MGDA direction solver in the authors' official `toy.py` implementation.[1] For $\theta = (\theta_1, \theta_2) \in \mathbb{R}^2$, the two objectives are

$$f_k(\theta) = c_1(\theta_2)\, h_k(\theta) + c_2(\theta_2)\, q_k(\theta), \qquad k \in \{1, 2\}, \tag{37}$$

where the gates $c_1(\theta_2) = \max\{\tanh(\theta_2/2), 0\}$ and $c_2(\theta_2) = \max\{\tanh(-\theta_2/2), 0\}$ select the upper and lower half-planes. In the upper half-plane, each objective is a logarithmic valley,

$$\begin{aligned} h_1(\theta) &= \log\max\{|0.5\theta_1 + 3.5 - \tanh\theta_2|,\, \epsilon\} + 6, \\ h_2(\theta) &= \log\max\{|0.5\theta_1 - 3.5 + \tanh\theta_2|,\, \epsilon\} + 6, \end{aligned} \tag{38}$$

with $\epsilon = 5 \times 10^{-6}$, and in the lower half-plane it is a quadratic bowl,

$$\begin{aligned} q_1(\theta) &= \left[(\theta_1 - 7)^2 + 0.1(\theta_2 + 8)^2\right]/10 - 20, \\ q_2(\theta) &= \left[(\theta_1 + 7)^2 + 0.1(\theta_2 + 8)^2\right]/10 - 20. \end{aligned} \tag{39}$$

Hence f_1 and f_2 are minimized near $(7, -8)$ and $(-7, -8)$, respectively, and the two objectives conflict along the segment between them. In Fig. 1, the leftmost panel plots the mean loss $\frac{1}{2}(f_1 + f_2)$ as a surface; the top row of the 2×2 block shows the contours of f_1 and f_2, and the bottom row shows the MGDA and MGDA+smoothing trajectories on the contours of the mean loss. The MGDA baseline follows the original implementation without changing the objective functions or direction solver. Both trajectory panels use Adam with learning rate 10^{-3} for 100,000 updates, starting at $(-8.5, 7.5)$, $(-8.5, -5.0)$, and $(9.0, 9.0)$, as in the released script.

The original toy problem uses exact gradients. To mimic the stochastic gradient estimates in diffusion RL, we perturb each objective gradient at every update with independent Gaussian noise, $\widehat{g}_k = g_k + \xi_k$ with $\xi_k \sim \mathcal{N}(0, \sigma^2 I)$ and $\sigma = 0.2$. Both methods use the same noise draws at each initialization. Objective-wise EMA is the only algorithmic difference: MGDA uses the current estimates directly ($\mu = 0$), while MGDA+smoothing maintains a separate zero-initialized EMA per objective with $\mu = 0.7$ and applies bias correction before the same MGDA solve. These paired toy trajectories illustrate the effect of smoothing in this setting; they do not evaluate the complete STB algorithm or establish convergence of diffusion training. Black dots mark initializations, and trajectory colors progress from red to gold with the update index. We plot every 100th iterate without post-hoc filtering of the parameter paths. The gray reference marks a numerically verified lower Pareto-stationary branch, not a claim to characterize the entire global Pareto front. Stars indicate individual-objective minima in the objective panels and the equal-weight stationary solution in the trajectory panels.

D ADDITIONAL QUALITATIVE COMPARISONS

Figs. 6, 7, and 8 show additional samples on DrawBench, GenEval, and OCR prompts, respectively. All four methods use the same prompt in each row. Key constraints in each prompt are highlighted in red, and long OCR prompts are abbreviated for display.

[1] `https://github.com/Cranial-XIX/CAGrad/blob/main/toy.py`

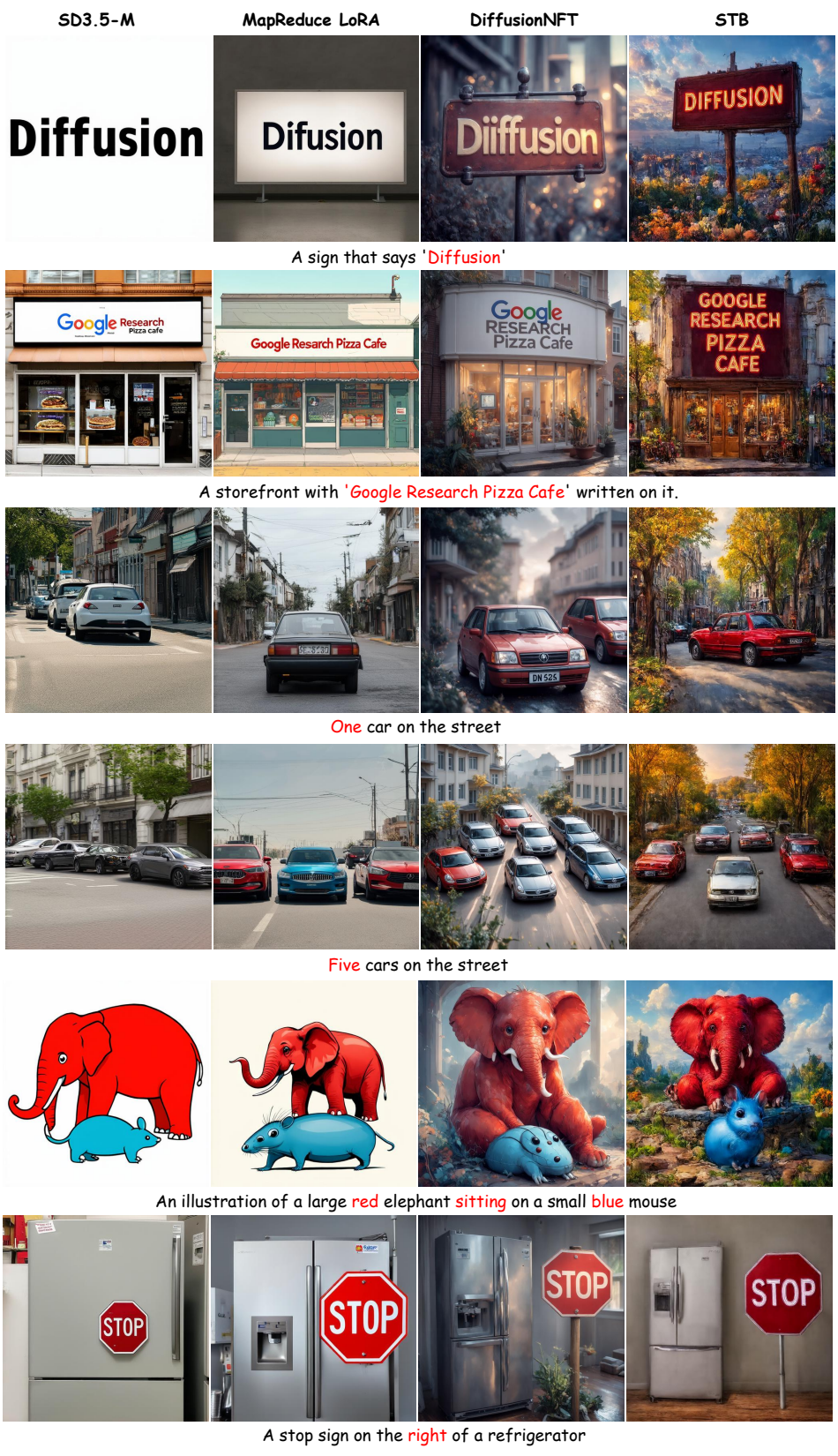


Figure 6: **Qualitative comparison on DrawBench.** Samples from SD3.5-M, MapReduce LoRA, DiffusionNFT, and STB for the same prompts, with key constraints highlighted in red.



Figure 7: **Qualitative comparison on GenEval.** Samples from SD3.5-M, MapReduce LoRA, DiffusionNFT, and STB for the same prompts, with key constraints highlighted in red.

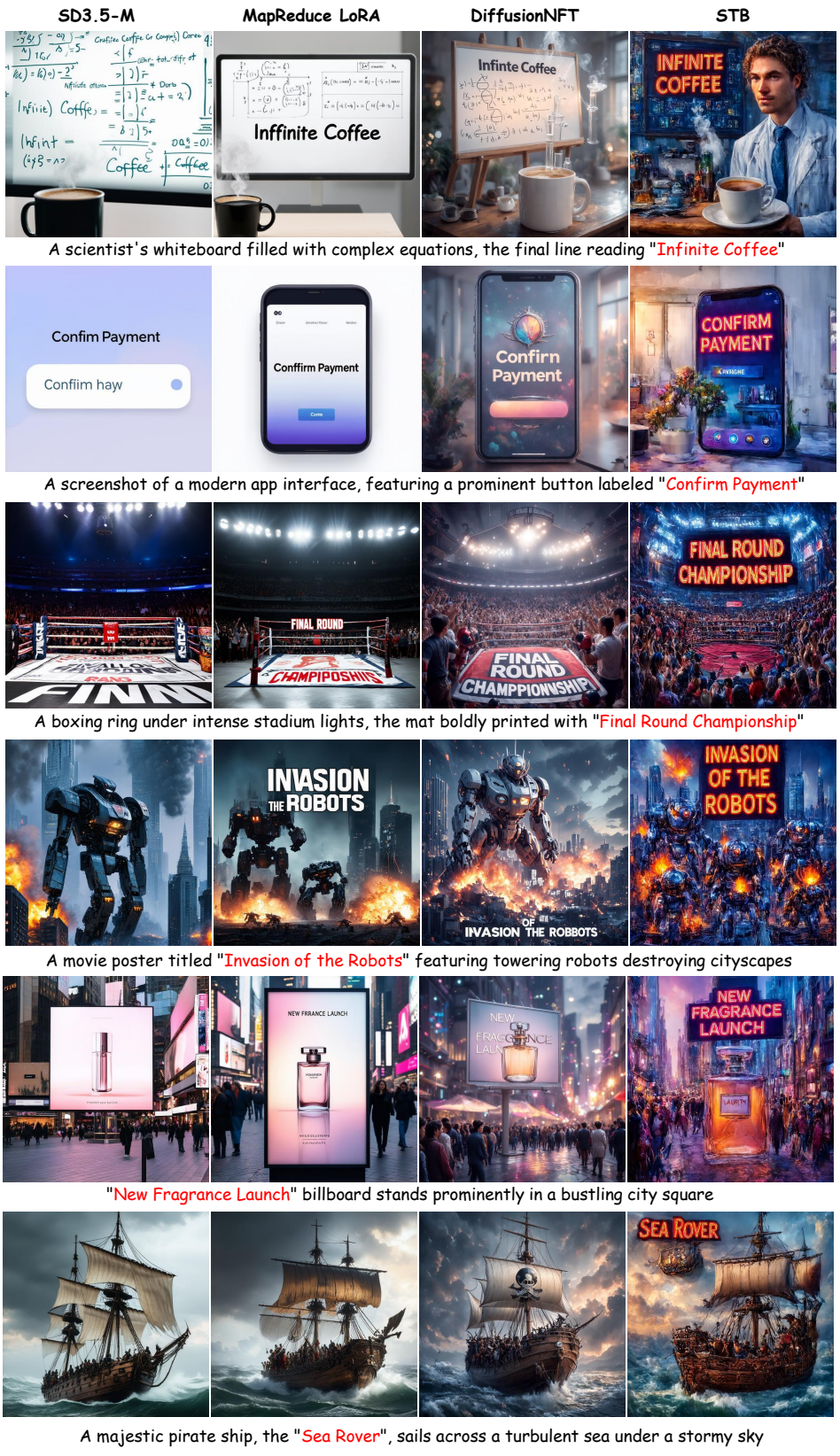


Figure 8: **Qualitative comparison on OCR.** Samples from SD3.5-M, MapReduce LoRA, DiffusionNFT, and STB for the same prompts, with the text to be rendered highlighted in red.