PSTUNE: TEXT-TO-IMAGE REWARD UPDATES THAT TRANSFER ONLY WHEN PROMPT AND STATE MATCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Reward fine-tuning raises a text-to-image model's score under a learned preference reward but often lowers what the reward does not measure: tuning a LoRA on FLUX.1-dev with HPSv2.1 reduces GenEval accuracy from 67.5% to 63.1% and within-prompt LPIPS from 54.1 to 46.6. We trace this loss to how a reward update spreads. The gradient from one image changes weights that every image uses, and its first-order effect on another image is set by the cross-Jacobian of the two generations. For an adapter that reads concatenated prompt and state features, this cross-Jacobian is the *sum* of a prompt similarity and a state similarity, so the update reaches any image that matches in either; it should reach an image only as far as both match. PSTune builds this rule into one three-way tensor that maps the normalized prompt embedding and the centered latent–velocity feature at each location to a velocity correction for a frozen generator. With detached features and scalar SGD, the change one image's update causes in another factors exactly into prompt similarity, state overlap, and the source gradient, and centering removes the image-wide gradient term that would bypass this product. On FLUX.1-dev, SD3.5-Medium, and SDXL, PSTune raises GenEval ImageReward over matched LoRA by 0.29–0.31 with direct training and 0.20–0.25 with policy gradients, at the frozen model's accuracy and LPIPS.

1 INTRODUCTION

Reward fine-tuning adapts a pretrained text-to-image generator so that its images score higher under a learned preference model such as HPSv2.1 or ImageReward (Wu et al., 2023; Xu et al., 2023), using gradients through the sampler (Clark et al., 2024; Prabhudesai et al., 2023) or policy-gradient estimates (Black et al., 2024; Liu et al., 2025). The trained score rises, but capabilities that the reward does not measure often fall. After 1,024 updates of a rank-32 LoRA on HPSv2.1, FLUX.1-dev gains 3.2 HPS points, yet its GenEval accuracy drops from 67.5% to 63.1% and its within-prompt LPIPS from 54.1 to 46.6 (Table 1): fewer requested attributes appear, and samples of one prompt look more alike.

We attribute this loss to where a reward update lands. Suppose HPS prefers the glossy finish of an image of "a red sports car at sunset", so its gradient strengthens specular highlights on the car's smooth, bright regions. The LoRA weights that carry this change are read by every prompt at every step (Figure 1), so the push also reaches "a cardboard rocket", whose smooth surfaces look similar although its request is matte cardboard, and the sky of the car image, whose request matches although its content differs. Existing remedies change which gradient is computed, through KL penalties, clipped ratios, or diversity-aware and direction-decoupled objectives (Fan et al., 2023; Wang et al., 2026; Liu et al., 2026; Chen et al., 2026), and still apply it to parameters that all generations share.

We make this spread an explicit design choice. To first order, a gradient on a *source* image i changes the output on a *target* image j through the cross-Jacobian of the two generations, the adapter's empirical tangent kernel (Jacot et al., 2018). For an adapter that is linear in concatenated prompt and state features $[\boldsymbol{p}; \boldsymbol{s}]$, this kernel is the sum $\boldsymbol{p}_j^\top \boldsymbol{p}_i + \boldsymbol{s}_j^\top \boldsymbol{s}_i$, and the update reaches a target that resembles the source in *either* prompt or state. For an adapter that is linear in their outer product $\boldsymbol{p} \otimes \boldsymbol{s}$, the kernel is the product $(\boldsymbol{p}_j^\top \boldsymbol{p}_i)(\boldsymbol{s}_j^\top \boldsymbol{s}_i)$, and the update reaches a target only as far as it resembles the source in *both*. The car's gradient then reaches the rocket in proportion to prompt

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.

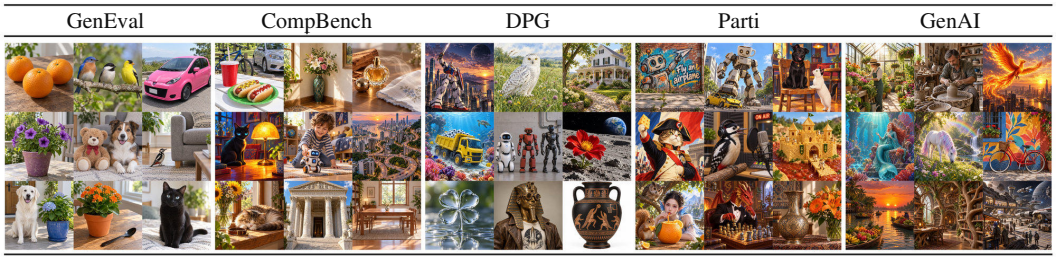


Figure 1: **One set of adapted weights serves very different requests.** PSTune outputs for GenEval, T2I-CompBench, DPG-Bench, PartiPrompts, and GenAI-Bench prompts. A reward update computed on any one image is read by all others.

similarity and the sky in proportion to state similarity. One path bypasses the product: the image-wide part of a gradient, such as raising contrast everywhere, pairs with the spatial mean of the state features and reaches every prompt-matched target; subtracting that mean removes it.

PSTune (*prompt–state tuning*) implements both rules with one tensor $\boldsymbol{W} \in \mathbb{R}^{d_v \times d_c \times d_s}$ at the velocity output of a frozen FLUX.1-dev, SD3.5-Medium, or SDXL model: the normalized pooled text embedding contracts $\boldsymbol{W}$ into a prompt-specific matrix, which maps the centered latent–velocity feature at each location and step to a velocity correction. Forward pass and gradient share the detached features, so a scalar SGD step changes target j's correction by exactly $-\eta\,(\boldsymbol{p}_j^\top \boldsymbol{p}_i)\,\boldsymbol{S}_j \boldsymbol{S}_i^\top \boldsymbol{G}_i$ at fixed states (Theorem 1), with 6.3M parameters on FLUX.1-dev against 131M for LoRA.

PSTune improves GenEval ImageReward over matched LoRA by 0.29–0.31 with direct training and 0.20–0.25 with Flow-GRPO on three backbones, with accuracy and LPIPS at the frozen model's level (Figure 2). At equal update norm, updates stored under a wrong prompt or state index still raise their own image's reward but lower ImageReward on other images.

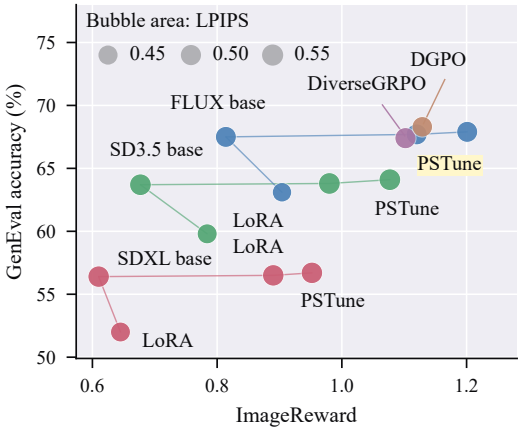


Figure 2: **LoRA buys reward with accuracy and variety; PSTune keeps both.** GenEval accuracy against ImageReward (three-seed means); bubble area is LPIPS. Lines join each base with LoRA or PSTune after 512 and 1,024 direct updates.

2 Related Work

Reward fine-tuning of text-to-image models. Early methods reweight the likelihood by reward (Lee et al., 2023) or backpropagate a differentiable reward through the sampler (Xu et al., 2023; Clark et al., 2024; Prabhudesai et al., 2023); treating denoising as a decision process enabled policy gradients (Black et al., 2024; Fan et al., 2023), and preference pairs replaced explicit rewards (Wallace et al., 2024; Yang et al., 2024; Zhu et al., 2025). For flow models, Flow-GRPO and DanceGRPO use group-relative policy gradients (Liu et al., 2025; Xue et al., 2025), DiffusionNFT optimizes on the forward process (Zheng et al., 2026), and DGPO learns from group preferences (Luo et al., 2026). Reward over-optimization and diversity loss are documented in language models (Gao et al., 2023; Kirk et al., 2024), with related failures in diffusion fine-tuning (Wang et al., 2026; Liu et al., 2026). Responses reshape the objective through KL regularization (Fan et al., 2023; Liu et al., 2025), regulated clipping (Wang et al., 2026), calibrated or sharpness-aware rewards (Kim et al., 2024; Kim & Sim, 2026), diversity-aware advantages (Liu et al., 2026), or a decoupled preference direction (Chen et al., 2026). All of them decide which gradient each image produces and apply it to full weights or LoRA, which every prompt and step reads, so how far a gradient spreads is left to the parameterization. PSTune keeps these objectives and sets this spread explicitly.

How the update is parameterized. LoRA and DoRA adapt shared model weights using low-rank components (Hu et al., 2022; Liu et al., 2024). Input-dependent parameters condition the change on the input: hypernetworks generate weights from a conditioning vector (Ha et al., 2017; Ruiz et al., 2024), FiLM scales and shifts features (Perez et al., 2018), and HyperAlign generates low-rank weights from prompt and diffusion-state information (Xie et al., 2026). Bilinear and multiplicative models show that a product of two inputs expresses interactions that a sum cannot (Tenenbaum & Freeman, 2000; Jayakumar et al., 2020). These designs target what the network can represent. How a gradient on one input changes the output on another is given by the tangent kernel (Jacot et al., 2018), which in these designs arises implicitly from the generated weights' Jacobians; in reward fine-tuning, this kernel decides which images receive each reward gradient. PSTune reads fixed prompt and state features through a bilinear tensor, so its tangent kernel is the product of prompt and state similarities.

3 Where Should a Reward Update Go?

3.1 How an update on one image changes another

A flow-based generator integrates a velocity field over K native steps, $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + h_k \boldsymbol{v}^0(\boldsymbol{x}_k, t_k, c)$, from noise $\boldsymbol{x}_0$ under prompt c (Lipman et al., 2023; Liu et al., 2023; Esser et al., 2024); SDXL's sigma-Euler sampler has the same form (Appendix C.1). Reward fine-tuning adds a trainable correction $\boldsymbol{u}_\theta$ to the velocity and minimizes a reward loss $\mathcal{L}$, direct or policy-gradient. A generation whose gradient enters an update is a *source* i; one that later runs with the updated parameters is a *target* j. We stack generation i's corrections over all locations and steps into $\boldsymbol{U}_i$ and write $\boldsymbol{J}_i = \partial \operatorname{vec}(\boldsymbol{U}_i)/\partial\theta$ and $\boldsymbol{g}_i = \operatorname{vec}(\partial\mathcal{L}/\partial \boldsymbol{U}_i)$. With the inputs of both generations held fixed, a gradient step of size η changes the target's correction by

$$\operatorname{vec}(\Delta \boldsymbol{U}_{j\leftarrow i}) = -\eta\, \boldsymbol{J}_j \boldsymbol{J}_i^\top \boldsymbol{g}_i + \mathcal{O}(\eta^2). \tag{1}$$

The cross-Jacobian $\boldsymbol{J}_j \boldsymbol{J}_i^\top$ decides which targets move and in which direction. For LoRA and full fine-tuning it depends on the backbone's hidden activations, and nothing ties it to whether the two images request the same content. PSTune chooses the adapter to prescribe this matrix.

3.2 A sum of similarities versus a product

For an adapter that is linear in an input feature, $\boldsymbol{u} = \boldsymbol{\Theta}\boldsymbol{\phi}$ with trainable $\boldsymbol{\Theta}$, the cross-Jacobian between inputs $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}_j$ is $(\boldsymbol{\phi}_j^\top \boldsymbol{\phi}_i)\, \boldsymbol{I}$. With a prompt vector $\boldsymbol{p}$ and a local state vector $\boldsymbol{s}$, the two direct ways to combine them give

$$\boldsymbol{\phi} = [\boldsymbol{p}; \boldsymbol{s}]: \quad \boldsymbol{\phi}_j^\top \boldsymbol{\phi}_i = \boldsymbol{p}_j^\top \boldsymbol{p}_i + \boldsymbol{s}_j^\top \boldsymbol{s}_i, \qquad \boldsymbol{\phi} = \boldsymbol{p} \otimes \boldsymbol{s}: \quad \boldsymbol{\phi}_j^\top \boldsymbol{\phi}_i = (\boldsymbol{p}_j^\top \boldsymbol{p}_i)(\boldsymbol{s}_j^\top \boldsymbol{s}_i). \tag{2}$$

With concatenation, a matching state carries a source update to an unrelated prompt, and a matching prompt carries it to unrelated regions. With the outer product, the update is scaled by both similarities at once, and a small value of either one suppresses it. Reward fine-tuning needs the second, because a preferred appearance helps another image only where both requests and local states agree. A hypernetwork mapping $(\boldsymbol{p}, \boldsymbol{s})$ to weights adds its own Jacobian to this kernel; the product holds exactly for an adapter linear in $\boldsymbol{p} \otimes \boldsymbol{s}$.

3.3 The image-wide term

The product acts location by location, while the state features of one image share a spatial mean. At one image and step, let $\boldsymbol{F} \in \mathbb{R}^{N \times d_s}$ stack the raw state features of the N latent locations and $\boldsymbol{G} \in \mathbb{R}^{N \times d_v}$ the gradients of their corrections, with column means $\bar{\boldsymbol{f}} = \boldsymbol{F}^\top \mathbf{1}/N$ and $\bar{\boldsymbol{g}} = \boldsymbol{G}^\top \mathbf{1}/N$. With the centering matrix $\boldsymbol{C} = \boldsymbol{I}_N - \mathbf{1}\mathbf{1}^\top/N$, the gradient that a linear adapter receives splits as

$$\boldsymbol{G}^\top \boldsymbol{F} = \underbrace{N\, \bar{\boldsymbol{g}}\, \bar{\boldsymbol{f}}^\top}_{\text{image-wide term}} + \underbrace{(\boldsymbol{C}\boldsymbol{G})^\top (\boldsymbol{C}\boldsymbol{F})}_{\text{zero-mean term}}. \tag{3}$$

The first term pairs the average gradient, such as a uniform increase in contrast, with the average state; a target location reads it through its overlap with $\bar{\boldsymbol{f}}$, so it reaches every prompt-matched image regardless of local content. Replacing $\boldsymbol{F}$ by $\boldsymbol{C}\boldsymbol{F}$ removes it and leaves the zero-mean term unchanged (Theorem 2).

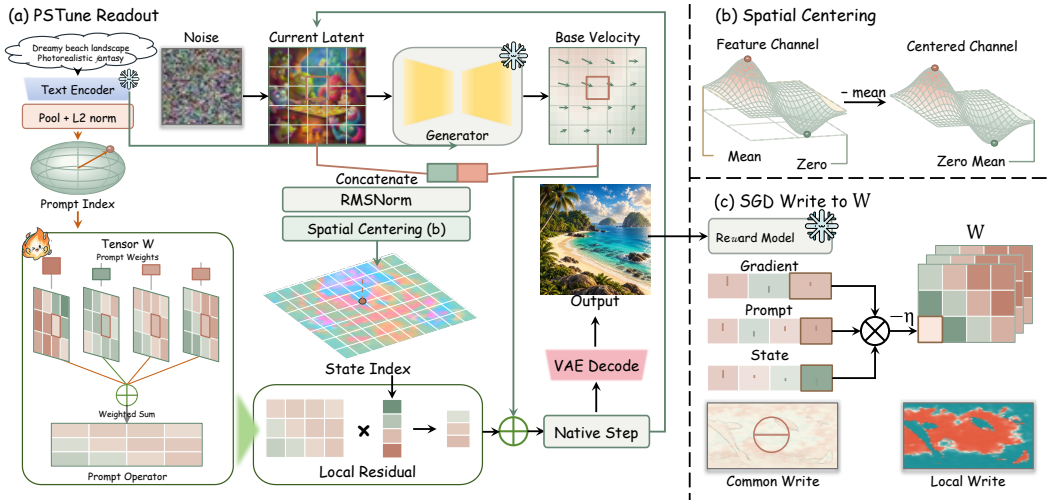


Figure 3: **PSTune.** (a) The prompt index contracts W into a prompt operator, which maps the state index at each location to a correction of the frozen velocity. (b) Centering subtracts the per-image, per-step mean of each feature channel. (c) Scalar SGD updates W through the same indices, so the change at another image is the product in Eq. (7).

4 PSTune

Section 3 asks for three properties: a source update reaches a target in proportion to the product of prompt and state similarities (Eq. (2)), the image-wide term of Eq. (3) is absent, and both hold exactly at every update (Figure 3).

4.1 Prompt and state indices

Prompt index. One vector must summarize a request, with an inner product that measures how similar two requests are. We normalize the pooled text embedding $\boldsymbol{q}(c)$ that each backbone already computes, $\boldsymbol{p}(c) = \boldsymbol{q}(c)/\|\boldsymbol{q}(c)\|_2 \in \mathbb{R}^{d_c}$, so that $\boldsymbol{p}_j^\top \boldsymbol{p}_i$ is a cosine similarity (Appendix C.1). Token-level embeddings vary in length and have no single inner product; the denoiser still receives them natively.

State index. The state vector must describe what a location contains and where the frozen model is moving it, so that two locations count as similar only when both agree. At step k and location n we concatenate the latent $\boldsymbol{x}_{k,n} \in \mathbb{R}^{d_v}$ and the frozen velocity after guidance $\boldsymbol{v}^0_{k,n} \in \mathbb{R}^{d_v}$. Their scales vary across noise levels and backbones, so we normalize them without learned parameters, stop the gradient so that the feature is a fixed index, and subtract the per-image, per-step spatial mean:

$$\boldsymbol{f}_{k,n} = \mathrm{sg}\big(\,\mathrm{RMSNorm}([\boldsymbol{x}_{k,n}; \boldsymbol{v}^0_{k,n}])\big) \in \mathbb{R}^{d_s}, \qquad \boldsymbol{s}_{k,n} = \boldsymbol{f}_{k,n} - \frac{1}{N}\sum_{m=1}^{N} \boldsymbol{f}_{k,m}, \tag{4}$$

where $d_s = 2d_v$, $\mathrm{RMSNorm}(\boldsymbol{z}) = \boldsymbol{z}/(\|\boldsymbol{z}\|_2^2/d_s + \epsilon)^{1/2}$, and $\epsilon = 2^{-23}$. The index follows the actual latent at every step.

4.2 A three-way tensor reads the product

The smallest linear map with input $\boldsymbol{p} \otimes \boldsymbol{s}$ is a three-way tensor $\boldsymbol{W} \in \mathbb{R}^{d_v \times d_c \times d_s}$; contracting its prompt mode gives one matrix $\boldsymbol{A}(c)$ per request, which maps each state index to a velocity correction:

$$\boldsymbol{A}(c) = \boldsymbol{W} \times_2 \boldsymbol{p}(c) \in \mathbb{R}^{d_v \times d_s}, \qquad \boldsymbol{u}_{k,n} = \boldsymbol{A}(c)\, \boldsymbol{s}_{k,n}, \qquad \boldsymbol{v}_{k,n} = \boldsymbol{v}^0_{k,n} + \boldsymbol{u}_{k,n}, \tag{5}$$

where $\times_2$ denotes the mode-2 product, $A_{ab} = \sum_h W_{ahb}\, p_h$ (Kolda & Bader, 2009). Zero initialization recovers the frozen model. The correction is added after native guidance, so each model keeps

its sampler; $\boldsymbol{A}(c)$ is computed once per prompt, and the tensor has 6.3M parameters on FLUX.1-dev, 1.0M on SD3.5-Medium, and 41k on SDXL (Table 7).

4.3 The update is an exact product

Let $\boldsymbol{g}_{i,k,n} = \partial\mathcal{L}/\partial\boldsymbol{u}_{i,k,n}$ be the gradient that the training graph assigns to a correction, including batch averaging and every downstream path through the frozen sampler. Because the indices are detached, each correction contributes one rank-one term, and a scalar SGD step gives

$$\boldsymbol{W}^{+} = \boldsymbol{W} - \eta \sum_{i,k,n} \boldsymbol{g}_{i,k,n} \otimes \boldsymbol{p}_i \otimes \boldsymbol{s}_{i,k,n}. \tag{6}$$

We stack generation i's state indices into $\boldsymbol{S}_i \in \mathbb{R}^{M_i \times d_s}$ and its gradients into $\boldsymbol{G}_i \in \mathbb{R}^{M_i \times d_v}$, where M_i counts its locations over the steps that receive gradient; its corrections are $\boldsymbol{U}_i = \boldsymbol{S}_i \boldsymbol{A}(c_i)^{\top}$.

Theorem 1 (Prompt–state factorization)**.** *Under the update in Eq. (6), the contribution of source i changes the correction of target j, evaluated at the target's current indices, by*

$$\Delta\boldsymbol{U}_{j\leftarrow i} = -\eta \quad \underbrace{(\boldsymbol{p}_j^{\top}\boldsymbol{p}_i)}_{\text{prompt similarity}} \quad \underbrace{\boldsymbol{S}_j\boldsymbol{S}_i^{\top}}_{\text{state overlap}} \quad \boldsymbol{G}_i. \tag{7}$$

The identity is exact for any finite η, and the contributions of several sources add.

Theorem 2 (Removal of the image-wide term)**.** *For fixed $\boldsymbol{F}$ and $\boldsymbol{G}$, $\boldsymbol{G}^{\top}\boldsymbol{C}\boldsymbol{F} = (\boldsymbol{C}\boldsymbol{G})^{\top}(\boldsymbol{C}\boldsymbol{F})$. A spatially constant gradient therefore produces no update, and a zero-mean gradient the same update as with raw features; $\boldsymbol{C}$ is the only fixed linear map with both properties.*

Reading the two theorems. Theorem 1 follows by contracting the source's term in Eq. (6) with $\boldsymbol{p}_j$ and $\boldsymbol{S}_j$ (Appendix B); since the features are fixed functions of the current generation, it realizes the product of Eq. (2) at every training step. The prompt factor sets how strongly and in which direction a target moves (Figure 8, Appendix B.2), and the state factor sets where: in Figure 4, the same update stored under another image's state index acts on different content of the scene at the same strength.

Why a scalar step. A coordinate-wise preconditioner such as Adam or RMSProp (Kingma & Ba, 2015) scales the entries of Eq. (6) differently, so a target with a prompt orthogonal to the source still changes, and at equal norm the response departs from the product (Appendix B.6, Figure 12). We therefore train $\boldsymbol{W}$ with SGD without momentum or weight decay; global gradient-norm clipping rescales the whole update and preserves Eq. (7).

The cost of centering. Centered features also give centered corrections, $\mathbf{1}^{\top}\boldsymbol{U} = \mathbf{1}^{\top}\boldsymbol{C}\boldsymbol{F}\boldsymbol{A}^{\top} = \mathbf{0}$, so global changes such as overall brightness come from the frozen model. We therefore vary the two uses of centering separately and write F/CF for raw forward and centered update features; PSTune is CF/CF.

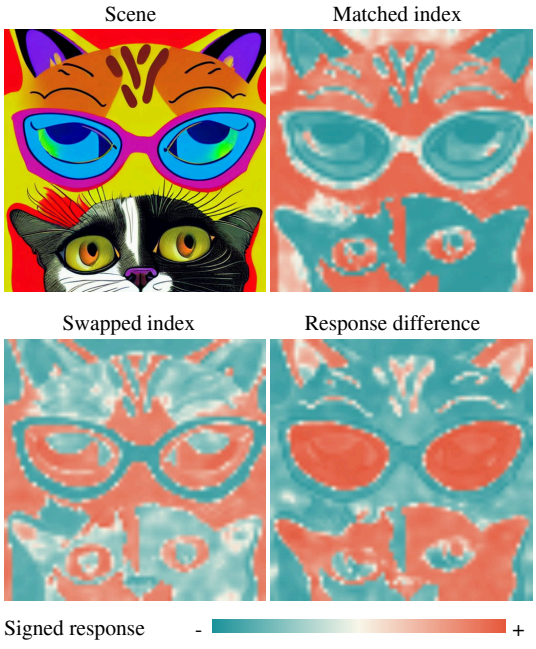


Figure 4: **The state index decides where an update acts.** Response of one source update stored under the matched and under a swapped state index, and their difference, at fixed gradient, readout, and norm.

From fixed states to a regenerated image. A regenerated target also recomputes its indices, which adds $(\boldsymbol{S}_j^{+} - \boldsymbol{S}_j)(\boldsymbol{A}_j^{+})^{\top}$ to Eq. (7); Appendix B.5 bounds it, and Section 5.3 measures it.

Table 1: FLUX.1-dev (12B, Black Forest Labs, 2024) and SD3.5-Medium (2.5B, guidance 4.5, Stability AI, 2024) on five benchmarks; three-seed means. IR: ImageReward; Pick: PickScore; VQA: VQAScore; Acc., VQA, LPIPS $\times 100$. Bold: block maxima (red in long blocks); blue: PSTune. Budgets in Table 8.

Method	HPS	GenEval					DPG-Bench		GenAI-Bench		PartiPrompts	
	HPS	HPS	IR	Pick	Acc.	LPIPS	Score	IR	VQA	IR	VQA	IR
FLUX.1-dev (12B) — direct pair and external native schedules												
Frozen base	31.44	28.64	0.814	22.213	67.5	54.14	88.47	1.110	74.10	1.181	77.87	1.046
LoRA (Hu et al., 2022), direct	34.62	31.31	0.904	22.475	63.1	46.63	84.28	1.265	71.07	1.243	74.02	1.174
Flow-GRPO (Liu et al., 2025)	**35.22**	**31.61**	1.004	22.821	64.9	49.70	86.08	1.334	72.78	1.304	75.58	1.247
GRPO-Guard (Wang et al., 2026)	34.18	30.86	1.071	22.850	67.1	52.18	88.22	1.390	73.76	1.405	77.48	1.346
DiverseGRPO (Liu et al., 2026)	33.98	30.57	1.102	22.947	67.4	**55.08**	88.44	1.387	74.35	1.427	78.03	1.341
DiffusionNFT (Zheng et al., 2026)	34.41	31.09	1.094	23.006	67.8	52.63	88.89	1.424	74.52	1.369	78.39	1.334
RSA-FT (Kim & Sim, 2026)	33.51	30.33	1.050	22.740	67.4	52.84	87.71	1.357	73.52	1.332	77.12	1.276
D2-Align (Chen et al., 2026)	33.85	30.24	1.078	22.863	66.8	54.56	88.29	1.440	74.00	1.396	77.75	1.377
DGPO (Luo et al., 2026)	34.53	31.10	1.129	**23.056**	**68.3**	51.87	**89.36**	1.418	**74.76**	1.400	**78.70**	1.333
PSTune, direct	33.55	30.47	**1.201**	22.925	67.9	54.27	88.60	**1.545**	74.20	**1.526**	78.00	**1.487**
FLUX.1-dev (12B) — matched policy-gradient objective												
LoRA, PG (Hu et al., 2022)	**35.05**	**31.89**	0.944	22.792	64.5	48.26	85.00	1.281	72.07	1.290	75.08	1.173
PSTune, PG	33.91	30.89	**1.158**	**23.034**	**68.1**	**54.43**	**88.73**	**1.517**	**74.27**	**1.539**	**78.14**	**1.418**
SD3.5-Medium (2.5B) — direct pair and external native schedules												
Frozen base	30.44	27.94	0.677	21.918	63.7	57.85	84.11	0.971	70.66	1.028	73.94	0.901
LoRA (Hu et al., 2022), direct	34.10	30.84	0.784	22.209	59.8	49.23	79.21	1.064	66.86	1.178	69.61	1.029
Flow-GRPO (Liu et al., 2025)	**34.55**	**31.13**	0.892	22.526	61.7	52.73	81.83	1.243	68.79	1.260	71.53	1.126
GRPO-Guard (Wang et al., 2026)	33.51	30.32	0.981	22.609	63.7	55.98	83.77	1.325	70.42	1.339	73.65	1.216
DiverseGRPO (Liu et al., 2026)	33.29	30.10	1.010	22.559	63.9	**58.89**	84.27	1.281	71.09	1.316	74.27	1.224
RSA-FT (Kim & Sim, 2026)	32.59	29.79	0.967	22.431	63.2	56.73	83.19	1.248	70.12	1.293	73.39	1.181
D2-Align (Chen et al., 2026)	33.07	29.89	0.985	22.486	63.3	57.84	84.02	1.297	70.65	1.310	73.77	1.247
DGPO (Luo et al., 2026)	33.91	30.43	1.012	**22.694**	**64.8**	55.39	**85.44**	1.317	**71.66**	1.345	**74.98**	1.229
PSTune, direct	32.88	29.89	**1.077**	22.550	64.1	57.95	84.26	**1.432**	70.85	**1.439**	74.19	**1.331**
SD3.5-Medium (2.5B) — matched policy-gradient objective												
LoRA, PG (Hu et al., 2022)	**34.69**	**31.32**	0.845	22.482	61.3	51.16	80.67	1.148	68.31	1.139	70.88	1.065
PSTune, PG	33.22	30.43	**1.095**	**22.721**	**63.8**	**58.06**	**84.44**	**1.449**	**70.84**	**1.426**	**74.15**	**1.306**

Table 2: SDXL-base 1.0 (2.6B, guidance 5.0, Podell et al., 2024); notation as in Table 1. T2I-CompBench ($\times 100$): texture, spatial, non-spatial, and complex composition.

Method	HPS bench.		GenEval					T2I-CompBench					
	HPS	IR	HPS	IR	Pick	Acc.	LPIPS	Color	Shape	Text.	Spat.	N-sp.	Cplx.
SDXL-base 1.0 (2.6B) — direct pair and external native schedules													
Frozen base	30.62	0.931	27.57	0.610	21.975	56.4	56.78	59.1	46.3	54.0	19.0	30.1	32.1
LoRA (Hu et al., 2022), direct	33.14	1.033	29.87	0.645	22.111	52.0	48.89	53.5	42.6	48.0	17.2	28.3	28.4
Flow-GRPO (Liu et al., 2025)	**33.55**	1.100	**30.16**	0.763	22.320	54.6	52.01	56.2	44.9	51.5	18.9	29.7	30.5
GRPO-Guard (Wang et al., 2026)	32.82	1.161	29.56	0.806	22.372	56.1	55.14	58.6	46.1	53.4	19.1	29.9	32.1
DiverseGRPO (Liu et al., 2026)	32.59	1.141	29.37	0.804	22.480	56.6	**57.78**	59.1	47.1	**54.4**	18.8	**30.5**	33.0
RSA-FT (Kim & Sim, 2026)	32.51	1.201	29.37	0.839	22.525	56.8	55.92	**59.9**	46.7	**54.4**	19.3	29.9	32.9
D2-Align (Chen et al., 2026)	32.76	1.133	29.33	0.785	22.338	55.5	56.40	58.4	45.2	52.4	18.5	29.9	31.9
DGPO (Luo et al., 2026)	33.17	1.188	29.84	0.825	**22.568**	**57.3**	54.38	59.6	**47.2**	54.0	**20.3**	29.9	**33.7**
PSTune, direct	32.40	**1.267**	29.05	**0.952**	22.474	56.7	56.87	59.1	47.1	54.1	19.2	29.7	32.4
SDXL-base 1.0 (2.6B) — matched policy-gradient objective													
LoRA, PG (Hu et al., 2022)	**33.86**	1.046	**30.50**	0.706	22.374	53.8	51.07	55.3	44.4	50.6	18.6	29.5	30.1
PSTune, PG	32.76	**1.247**	29.58	**0.909**	**22.652**	**56.8**	**56.69**	**59.3**	**46.4**	**54.2**	**19.6**	**30.2**	**32.8**

5 EXPERIMENTS

We test whether PSTune keeps adherence and variety while raising preference (Section 5.1), whether the gain comes from its rule (Section 5.2), and whether an update's index decides if it helps other images (Section 5.3).

Models and training. We tune FLUX.1-dev, SD3.5-Medium, and SDXL-base 1.0 (Black Forest Labs, 2024; Stability AI, 2024; Podell et al., 2024) at 1024×1024 with native samplers and guidance (Table 7), optimizing HPSv2.1 (Wu et al., 2023) for 1,024 updates on 8,192 DiffusionDB prompts (Wang et al., 2023). Direct training backpropagates the reward through the final step with batches of 16 (16.4k images); the policy-gradient (PG) condition uses Flow-GRPO with groups of 24 ten-step

trajectories (24.6k images). The matched rank-32 LoRA (AdamW) shares prompts, image budgets, and a three-candidate learning-rate search scored by validation ImageReward. External methods train on the same prompts and reward with native schedules (Table 8).

Evaluation. ImageReward (IR) on held-out prompts measures preference with a model other than the training reward (Xu et al., 2023), GenEval accuracy measures requested objects, counts, colors, and positions (Ghosh et al., 2023), and within-prompt LPIPS measures variety (Zhang et al., 2018); DPG-Bench, GenAI-Bench, T2I-CompBench, PickScore, VQAScore, and blinded PartiPrompts judgments complete the picture (Hu et al., 2024; Li et al., 2024; Huang et al., 2023; Kirstain et al., 2023; Lin et al., 2024; Yu et al., 2022); tables report three-seed means.

5.1 Preference quality with retained adherence and variety

Direct PSTune has the highest GenEval ImageReward of all direct and external methods on every backbone: 1.201 on FLUX.1-dev, 1.077 on SD3.5-Medium, and 0.952 on SDXL, against 0.904, 0.784, and 0.645 for direct LoRA (Tables 1 and 2); the paired FLUX gain is 0.297 (95% CI: 0.270–0.324), without LoRA's losses. FLUX GenEval accuracy is 67.9% for PSTune, 67.5% for the base, and 63.1% for LoRA, and within-prompt LPIPS, which grows when samples of one prompt differ in color and layout (Figure 5), is 54.27, 54.14, and 46.63, with the same order on SD3.5 and SDXL. PSTune also stays at the base level on DPG-Bench and GenAI-Bench VQAScore (FLUX, SD3.5) and exceeds matched LoRA in all six T2I-CompBench categories on every backbone (Table 9, Appendix E.1).

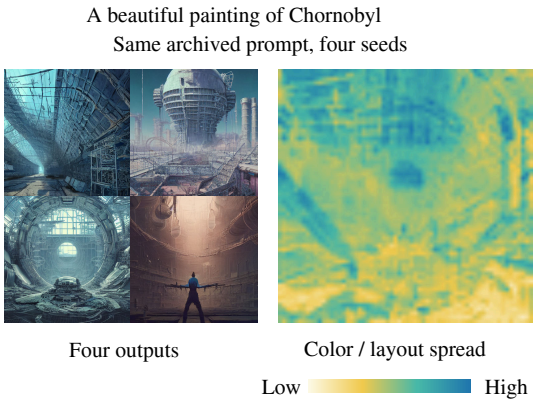


Figure 5: **What within-prompt variation measures.** Four samples of one prompt and their per-pixel CIELAB spread (Appendix D.1); brighter regions differ more in color or layout.

The training reward shows why. On FLUX, LoRA raises HPS by 3.18 points and ImageReward by 0.090, PSTune by 2.11 and 0.387, and Flow-GRPO reaches the highest HPS with a lower ImageReward than PSTune: a larger training-reward gain comes with a smaller independent-preference gain, as expected when HPS-specific changes spread to images where they do not fit. With policy gradients, PSTune improves ImageReward over PG LoRA by 0.203–0.250 with higher accuracy and LPIPS on all backbones, so the rule works with both gradient estimators (task-level results in Appendix E).

Table 3: FLUX PartiPrompts preferences, PSTune minus opponent in net-win points (Appendix D.2); positive favors PSTune. Train.: training images (k).

Opponent	PartiPrompts net wins (%)				Train.
	Match	Real.	Overall	Div.	k img.
LoRA (Hu et al., 2022)	**14.0**	**13.3**	**19.8**	**19.8**	16.4
Flow-GRPO (Liu et al., 2025)	8.2	10.2	12.8	7.9	2304.0
GRPO-Guard (Wang et al., 2026)	3.8	4.2	12.2	4.9	587.5
DiverseGRPO (Liu et al., 2026)	-0.7	8.3	11.6	-5.1	384.0
DiffusionNFT (Zheng et al., 2026)	-2.9	-5.7	10.5	3.0	2304.0
RSA-FT (Kim & Sim, 2026)	2.1	7.3	10.1	4.6	16.0
D2-Align (Chen et al., 2026)	0.7	1.9	13.1	-0.2	8.2
DGPO (Luo et al., 2026)	-3.5	-2.6	12.0	4.3	48.0

Human judgments. Blinded raters compare PartiPrompts images on text match, realism, overall preference, and cross-prompt diversity (Table 3). PSTune wins overall against all eight opponents, by 10.1 to 19.8 net-win points, and every question against LoRA. Its 19.8-point diversity lead over LoRA agrees with the LPIPS gap above. DiverseGRPO leads on diversity, and DiffusionNFT (140 times PSTune's training images) and DGPO on text match and realism. In Figure 6, LoRA turns a cardboard rocket on the moon into a cardboard airplane on a desk, while PSTune keeps the requested objects, materials, scenes, and text.

5.2 The gain comes from the prompt–state rule

LoRA differs from PSTune in location, size, and optimizer at once. To isolate the rule, we keep PSTune's tensor and forward pass and change only the update at equal norm (Table 4): another



Figure 6: **PSTune changes rendering and keeps the requested content.** Base, LoRA, and PSTune on the same requests.

batch sample's prompt index breaks the prompt factor of Eq. (7), its state sequence breaks the state factor, RMSProp breaks the scalar step, and raw update features restore the image-wide term.

Every change lowers ImageReward on all backbones: a wrong prompt index by 0.143–0.204, a wrong state index by 0.149–0.254, and RMSProp by 0.042–0.104, so the direction set by the index matters beyond the update's size. Centering the update features adds 0.226–0.276 with raw forward features ($F/F \rightarrow F/CF$) and 0.171–0.214 with centered ones ($CF/F \rightarrow CF/CF$), as much as a correct index. Without centering, the tensor (F/F) lands close to LoRA (0.923 versus 0.904 on FLUX), while both wrong-index controls, which keep centering, stay above LoRA. Centering and correct indices thus make separate contributions to the gain.

Table 4: Equal-shape controls at PSTune's update norm; three-seed means. F/CF: forward/update features. ΔM: PSTune minus control, FLUX PartiPrompts text match. Bold: IR maximum; in ΔM, the one control PSTune trails.

Adapter	GenEval ImageReward			Parti
	FLUX	SD3.5	SDXL	ΔM
Wrong prompt	0.997	0.934	0.777	8.4
Wrong state	1.052	0.851	0.698	5.8
F/F	0.923	0.823	0.606	9.7
F/CF	1.149	1.073	0.882	**-4.2**
CF/F	1.011	0.906	0.738	10.2
RMSProp	1.101	1.035	0.848	2.3
PSTune (CF/CF)	**1.201**	**1.077**	**0.952**	—

Removing the image-wide term alone. From each checkpoint at update 1,023, we compute one batch update twice from the same rollout, gradient, and step, with raw and with centered update features, so the pair differs only by the image-wide term of Eq. (3). Removing it raises target GenEval ImageReward on all backbones and forward branches, by 3.88–6.36×10^{-3} per update (Table 5), with PartiPrompts in the same direction (Figure 7c), while every FLUX source keeps a positive HPS gain (Figure 7a): the removed term mainly moved other images.

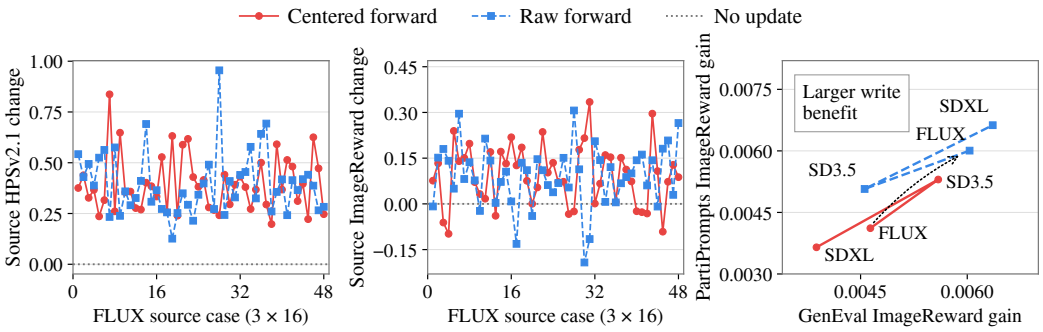
Table 5: Removing the image-wide term from one batch update: centered-minus-raw target GenEval ΔIR (10^{-3}), three final batches.

Forward	GenEval ΔIR (10^{-3})		
	FLUX	SD3.5	SDXL
Uncentered (F)	**6.03**	4.56	**6.36**
Centered (CF)	4.64	**5.59**	3.88

The cost of centering the forward correction. With centered updates, raw forward features (F/CF) lead PSTune by 4.2 net-win points of FLUX text match, and PSTune has higher ImageReward on all backbones (Table 4). Human judgments agree on all backbones: F/CF leads on text match by 4.2–6.9 points, PSTune on realism by 3.6–5.2 and overall by 9.2–11.7 (Table 11, Appendix F.2). Raw forward features let the tensor shift a whole image at once, which helps text match; PSTune's location-dependent corrections are preferred for realism and overall appeal.

5.3 The update index decides whether one image helps another

We apply one source's contribution to the checkpoint at update 1,024 and regenerate the source and its GenEval and PartiPrompts targets from the same noise. With correct indices, the source raises its own HPS and its targets' ImageReward on every backbone (Table 6).



(a) Source reward retention (b) Source quality response (c) Centered minus raw write

Figure 7: **Removing the image-wide term keeps source learning and helps other images.** (a,b) FLUX source HPS and ImageReward changes, 48 final-batch sources. (c) Centered-minus-raw target gains.

Table 6: Single-source updates at checkpoint 1,024. Write: index storing the update. Source ΔHPS (10^{-2}), target ΔIR (10^{-3}).

Write	Source ΔHPS	GenEval ΔIR	Parti ΔIR
FLUX.1-dev (12B)			
Correct	3.23	4.19	4.91
Wrong prompt	1.20	-2.80	-2.81
Wrong state	0.65	-3.45	-3.62
Cached state	**3.24**	**4.31**	**4.97**
SD3.5-Medium (2.5B)			
Correct	3.87	7.40	7.77
Wrong prompt	1.28	-3.53	-3.47
Wrong state	1.03	-2.35	-1.91
Cached state	**3.93**	**7.62**	**7.90**
SDXL-base 1.0 (2.6B)			
Correct	**3.56**	5.78	4.87
Wrong prompt	1.09	-4.02	-3.78
Wrong state	0.92	-2.67	-2.84
Cached state	**3.56**	**5.80**	**4.90**

A wrong prompt or state index still raises the source's HPS (20–37% of the correct gain on FLUX) but makes the target change negative on all backbones and both target lists; on FLUX/GenEval (10^{-3}), $+4.19$ becomes -2.80 (wrong prompt) and -3.45 (wrong state). Eq. (7) explains both effects: the source reads its update through the overlap of its correct and substituted index and keeps part of the gain, while each target reads an update stored for an unrelated prompt or state (Figure 4). The index sets the sign of transfer.

The fixed-state rule predicts regenerated outcomes. With the target's quality gradient $\boldsymbol{H}_j$, Eq. (7) gives the first-order change $-\eta(\boldsymbol{p}_j^\top \boldsymbol{p}_i)\langle \boldsymbol{H}_j, \boldsymbol{S}_j \boldsymbol{S}_i^\top \boldsymbol{G}_i \rangle_F$ (Proposition 2). On FLUX/GenEval at update 1,024, its sign matches the regenerated change for 95.4% of each source's targets, against 48.8% with a wrong prompt index (Figure 25). The prediction error grows over training, from a source-level RMSE of 0.00020 at initialization to 0.00208, because the indices and the frozen velocity follow the changed trajectory; the sign, which decides whether a target improves, remains predictable.

Recomputed indices. The cached-state arm of Table 6 keeps the old target indices while the latent evolves. Its mean target change stays within 0.22×10^{-3} of recomputation, yet per-image differences widen about 13-fold in both tails (Figure 26, Appendix F.4): recomputed indices redistribute the transfer across images.

6 Conclusion

Reward fine-tuning makes two decisions: which gradient each image produces, and which other images that gradient changes. Prior work designs the first; this paper designs the second. PSTune stores each update at the product of a prompt index and a centered state index, so the update reaches another image as far as both its request and its local denoising state match, and the image-wide term that would reach every image is removed. With at most 6.3M parameters and scalar SGD, this rule raises held-out preference quality on three backbones under two objectives at the frozen model's task accuracy and variety, and single-update interventions show that an update's index decides whether it helps other images. Because Eq. (1) holds for any adapter, the reach of an update can be measured and designed in the same way for other parameterizations.

AI USE STATEMENT

Generative AI assisted literature retrieval, drafting and polishing manuscript sections, mathematical derivation and proof checking, interpretation of results, and typesetting. The authors reviewed all AI-assisted work, checked the text and scientific claims, and take responsibility for the 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*, 2024. URL https://arxiv.org/abs/2305.13301.

Black Forest Labs. FLUX.1-dev model card. Model release, 2024. URL https://huggingface.co/black-forest-labs/FLUX.1-dev.

Chubin Chen, Sujie Hu, Jiashu Zhu, Meiqi Wu, Jintao Chen, Yanxun Li, Nisha Huang, Chengyu Fang, Jiahong Wu, Xiangxiang Chu, and Xiu Li. Taming Preference Mode Collapse via Directional Decoupling Alignment in Diffusion Reinforcement Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. URL https://arxiv.org/abs/2512.24146.

Kevin Clark, Paul Vicol, Kevin Swersky, and David Fleet. Directly Fine-Tuning Diffusion Models on Differentiable Rewards. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2309.17400.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling Rectified Flow Transformers for High-Resolution Image Synthesis. In *International Conference on Machine Learning*, 2024. URL https://proceedings.mlr.press/v235/esser24a.html.

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. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2305.16381.

Leo Gao, John Schulman, and Jacob Hilton. Scaling Laws for Reward Model Overoptimization. In *International Conference on Machine Learning*, 2023. URL https://arxiv.org/abs/2210.10760.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. GenEval: An object-focused framework for evaluating text-to-image alignment. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2310.11513.

David Ha, Andrew M. Dai, and Quoc V. Le. HyperNetworks. In *International Conference on Learning Representations*, 2017. URL https://arxiv.org/abs/1609.09106.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. ELLA: Equip Diffusion Models with LLM for Enhanced Semantic Alignment. arXiv preprint arXiv:2403.05135, 2024. URL https://arxiv.org/abs/2403.05135.

Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2I-CompBench: A Comprehensive Benchmark for Open-world Compositional Text-to-image Generation. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2307.06350v2.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural Tangent Kernel: Convergence and Generalization in Neural Networks. In *Advances in Neural Information Processing Systems*, 2018. URL https://arxiv.org/abs/1806.07572.

Siddhant M. Jayakumar, Wojciech M. Czarnecki, Jacob Menick, Jonathan Schwarz, Jack Rae, Simon Osindero, Yee Whye Teh, Tim Harley, and Razvan Pascanu. Multiplicative Interactions and Where to Find Them. In *International Conference on Learning Representations*, 2020. URL https://iclr.cc/virtual_2020/poster_rylnK6VtDH.html.

Kwanyoung Kim and Byeongsu Sim. Reward Sharpness-Aware Fine-Tuning for Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. URL https://arxiv.org/abs/2603.21175.

Kyuyoung Kim, Jongheon Jeong, Minyong An, Mohammad Ghavamzadeh, Krishnamurthy Dvijotham, Jinwoo Shin, and Kimin Lee. Confidence-aware Reward Optimization for Fine-tuning Text-to-Image Models. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2404.01863.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6980.

Robert Kirk, Ishita Mediratta, Christoforos Nalmpantis, Jelena Luketina, Eric Hambro, Edward Grefenstette, and Roberta Raileanu. Understanding the Effects of RLHF on LLM Generalisation and Diversity. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.06452.

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. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2305.01569.

Tamara G. Kolda and Brett W. Bader. Tensor Decompositions and Applications. *SIAM Review*, 51 (3):455–500, 2009. URL https://doi.org/10.1137/07070111X.

Kimin Lee, Hao Liu, Moonkyung Ryu, Olivia Watkins, Yuqing Du, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, and Shixiang Shane Gu. Aligning Text-to-Image Models using Human Feedback. arXiv preprint arXiv:2302.12192, 2023. URL https://arxiv.org/abs/2302.12192.

Baiqi Li, Zhiqiu Lin, Deepak Pathak, Jiayao Li, Yixin Fei, Kewen Wu, Tiffany Ling, Xide Xia, Pengchuan Zhang, Graham Neubig, and Deva Ramanan. Genai-bench: Evaluating and improving compositional text-to-visual generation. *CoRR*, abs/2406.13743, 2024.

Zhiqiu Lin, Deepak Pathak, Baiqi Li, Jiayao Li, Xide Xia, Graham Neubig, Pengchuan Zhang, and Deva Ramanan. Evaluating Text-to-Visual Generation with Image-to-Text Generation. In *European Conference on Computer Vision*, 2024. URL https://arxiv.org/abs/2404.01291.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow Matching for Generative Modeling. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.02747.

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*, 2026. URL https://arxiv.org/abs/2512.21514.

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. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2505.05470.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-Decomposed Low-Rank Adaptation. In *International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2402.09353.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2209.03003.

Yihong Luo, Tianyang Hu, and Jing Tang. Reinforcing Diffusion Models by Direct Group Preference Optimization. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=jymuXl8GYi.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual Reasoning with a General Conditioning Layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018. URL https://arxiv.org/abs/1709.07871.

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*, 2024. URL https://arxiv.org/abs/2307.01952.

Mihir Prabhudesai, Anirudh Goyal, Deepak Pathak, and Katerina Fragkiadaki. Aligning Text-to-Image Diffusion Models with Reward Backpropagation. arXiv preprint arXiv:2310.03739, 2023. URL https://arxiv.org/abs/2310.03739.

Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Wei Wei, Tingbo Hou, Yael Pritch, Neal Wadhwa, Michael Rubinstein, and Kfir Aberman. HyperDreamBooth: HyperNetworks for Fast Personalization of Text-to-Image Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. URL https://arxiv.org/abs/2307.06949.

Stability AI. Stable Diffusion 3.5 Medium model card. Model release, 2024. URL https://huggingface.co/stabilityai/stable-diffusion-3.5-medium.

Joshua B. Tenenbaum and William T. Freeman. Separating Style and Content with Bilinear Models. *Neural Computation*, 12(6):1247–1283, 2000. URL https://doi.org/10.1162/089976600300015349.

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*, 2024. URL https://arxiv.org/abs/2311.12908.

Jing Wang, Jiajun Liang, Jie Liu, Henglin Liu, Gongye Liu, Jun Zheng, Wanyuan Pang, Ao Ma, Zhenyu Xie, Xintao Wang, Meng Wang, Pengfei Wan, and Xiaodan Liang. GRPO-Guard: Mitigating Implicit Over-Optimization in Flow Matching via Regulated Clipping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. URL https://arxiv.org/abs/2510.22319.

Zijie J. Wang, Evan Montoya, David Munechika, Haoyang Yang, Benjamin Hoover, and Duen Horng Chau. DiffusionDB: A Large-scale Prompt Gallery Dataset for Text-to-Image Generative Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2023. URL https://aclanthology.org/2023.acl-long.51/.

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, 2023. URL https://arxiv.org/abs/2306.09341.

Xin Xie, Jiaxian Guo, and Dong Gong. HyperAlign: Hypernetwork for Efficient Test-Time Alignment of Diffusion Models. arXiv preprint arXiv:2601.15968, 2026. URL https://arxiv.org/abs/2601.15968.

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. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2304.05977.

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. URL https://arxiv.org/abs/2505.07818.

Kai Yang, Jian Tao, Jiafei Lyu, Chunjiang Ge, Jiaxin Chen, Weihan Shen, Xiaolong Zhu, and Xiu Li. Using Human Feedback to Fine-tune Diffusion Models without Any Reward Model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. URL https://doi.org/10.1109/CVPR52733.2024.00854.

Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, Ben Hutchinson, Wei Han, Zarana Parekh, Xin Li, Han Zhang, Jason Baldridge, and Yonghui Wu. Scaling Autoregressive Models for Content-Rich Text-to-Image Generation. arXiv preprint arXiv:2206.10789, 2022. URL https://arxiv.org/abs/2206.10789.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. URL https://arxiv.org/abs/1801.03924.

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*, 2026. URL https://arxiv.org/abs/2509.16117.

Huaisheng Zhu, Teng Xiao, and Vasant G. Honavar. DSPO: Direct Score Preference Optimization for Diffusion Model Alignment. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/7f70331dbe58ad59d83941dfa7d975aa-Abstract-Conference.html.

A Notation and Algorithm

Notation. We follow the notation of the ICLR template: scalars are italic (η, N), vectors bold lowercase ($\boldsymbol{p}$, $\boldsymbol{s}$, $\boldsymbol{g}$), matrices bold uppercase ($\boldsymbol{A}$, $\boldsymbol{S}$, $\boldsymbol{G}$), and the three-way tensor bold serif italic ($\boldsymbol{W}$). Local vectors are columns; a matrix that collects local vectors stacks them as rows, so at one image and step $\boldsymbol{F}, \boldsymbol{S} \in \mathbb{R}^{N\times d_s}$ and $\boldsymbol{U}, \boldsymbol{G} \in \mathbb{R}^{N\times d_v}$. The latent at step k is $\boldsymbol{x}_k$, and $\boldsymbol{X}_k$ arranges it with its N location vectors $\boldsymbol{x}_{k,n}$ as rows. The index $k = 0, \dots, K-1$ runs over sampler steps in execution order, with noise level t_k and step size h_k; $n = 1, \dots, N$ runs over latent locations; i and j denote a source and a target generation; r counts optimizer updates. The superscript 0 marks the frozen model and the superscript $+$ a quantity after one update. $\boldsymbol{A}(c) = \boldsymbol{W} \times_2 \boldsymbol{p}(c)$ is the prompt operator with entries $A_{ab} = \sum_h W_{ahb}\, p_h$, $\otimes$ is the outer product, $\langle \cdot, \cdot \rangle_F$ the Frobenius inner product, sg the stop-gradient operator, and $\boldsymbol{C} = \boldsymbol{I}_N - \mathbf{1}\mathbf{1}^\top/N$ the spatial centering matrix. In the tables, F and CF label raw and centered features, and F/CF lists the forward features before the update features.

Algorithm. Algorithm 1 states direct reward training and inference in the order they execute, following the input–initialization–update–output layout of Algorithm 1 in DiffusionNFT (Zheng et al., 2026). The trainable object and its update are those of Eqs. (5) and (6); Table 7 gives the dimensions and schedules, and Appendix C.2 the selection protocol.

Gradient and precision conventions. The prompt c enters the frozen model through its native text conditioning. Only the feature $\boldsymbol{f}_n$ is detached; the frozen velocity keeps its derivative with respect to $\boldsymbol{X}$ wherever the training graph retains that path. Lines 15–16 backpropagate direct training through the final sampler step, while the forward trajectory uses the current $\boldsymbol{W}$ at every step. Row n of $\boldsymbol{G}_i$ is $\boldsymbol{g}_{i,K-1,n}^\top$, and line 21 takes the norm over all tensor entries, so the effective step in Eq. (6) is $\eta\gamma$. The frozen components run in BF16; feature normalization, contractions, reductions, and $\boldsymbol{W}$ use FP32, and the correction is cast at the native interface.

Policy-gradient training. For matched PG training, Read supplies the velocity inside the Flow-GRPO transition, and the direct rollout and loss are replaced by groups of 24 ten-step stochastic

Algorithm 1 PSTune: direct reward training and inference with the native sampler

Require: frozen velocity after guidance $\boldsymbol{v}^0$, decoder Dec, pooled text encoder $\boldsymbol{q}$, reward R;
training prompts $\mathcal{D}$, noise distribution p_ξ, native step sizes $\{h_k\}_{k=0}^{K-1}$;
batch size $B = 16$, number of updates $J = 1{,}024$, selected learning rate η, $\epsilon = 2^{-23}$.
Initialize: $\boldsymbol{W} \leftarrow \boldsymbol{0} \in \mathbb{R}^{d_v \times d_c \times d_s}$ with $d_s = 2d_v$.

function $\mathrm{Read}(\boldsymbol{W}, c, \boldsymbol{p}, \boldsymbol{X}, k)$
  $\boldsymbol{V}^0 \leftarrow \boldsymbol{v}^0(\boldsymbol{X}, t_k, c)$ — frozen velocity after guidance, rows $\boldsymbol{v}_n^{0\top}$
  $\boldsymbol{z}_n \leftarrow [\boldsymbol{x}_n; \boldsymbol{v}_n^0], \quad n = 1, \dots, N$
  $\boldsymbol{f}_n \leftarrow \mathrm{sg}\Big(\boldsymbol{z}_n / \sqrt{\|\boldsymbol{z}_n\|_2^2/d_s + \epsilon}\Big), \quad n = 1, \dots, N$
  $\bar{\boldsymbol{f}} \leftarrow N^{-1} \sum_{n=1}^{N} \boldsymbol{f}_n; \quad \boldsymbol{s}_n \leftarrow \boldsymbol{f}_n - \bar{\boldsymbol{f}}$
  $\boldsymbol{A} \leftarrow \boldsymbol{W} \times_2 \boldsymbol{p}$ — computed once per prompt and $\boldsymbol{W}$
  $\boldsymbol{S} \leftarrow [\boldsymbol{s}_1, \dots, \boldsymbol{s}_N]^\top; \quad \boldsymbol{U} \leftarrow \boldsymbol{S}\boldsymbol{A}^\top; \quad \boldsymbol{V} \leftarrow \boldsymbol{V}^0 + \boldsymbol{U}$
  return $(\boldsymbol{V}, \boldsymbol{U}, \boldsymbol{S})$
end function
for $r = 0, \dots, J-1$ **do**
  $\{c_i\}_{i=1}^{B} \sim \mathcal{D}; \quad \xi_i \sim p_\xi$ independently
  $\boldsymbol{p}_i \leftarrow \boldsymbol{q}(c_i)/\|\boldsymbol{q}(c_i)\|_2; \quad \boldsymbol{X}_{i,0} \leftarrow \xi_i, \quad i = 1, \dots, B$
  for $k = 0, \dots, K-1$ **do**
    $(\boldsymbol{V}_{i,k}, \boldsymbol{U}_{i,k}, \boldsymbol{S}_{i,k}) \leftarrow \mathrm{Read}(\boldsymbol{W}, c_i, \boldsymbol{p}_i, \boldsymbol{X}_{i,k}, k), \quad \forall i$
    $\boldsymbol{X}_{i,k+1} \leftarrow \boldsymbol{X}_{i,k} + h_k \boldsymbol{V}_{i,k}, \quad \forall i$
    if $k < K-1$ **then** $\boldsymbol{X}_{i,k+1} \leftarrow \mathrm{sg}(\boldsymbol{X}_{i,k+1}), \quad \forall i$
  end for
  $I_i \leftarrow \mathrm{Dec}(\boldsymbol{X}_{i,K}); \quad \mathcal{L} \leftarrow -B^{-1} \sum_{i=1}^{B} R(I_i, c_i)$
  $\boldsymbol{G}_i \leftarrow \partial \mathcal{L} / \partial \boldsymbol{U}_{i,K-1}$ — includes the factor B^{-1}
  $\nabla \leftarrow \sum_{i=1}^{B} \sum_{n=1}^{N} \boldsymbol{g}_{i,K-1,n} \otimes \boldsymbol{p}_i \otimes \boldsymbol{s}_{i,K-1,n}$
  $\gamma \leftarrow 1/\max\{1, \|\nabla\|_F\}$ — one clipping factor for the batch
  $\boldsymbol{W} \leftarrow \boldsymbol{W} - \eta\gamma\nabla$ — scalar SGD, FP32 accumulation
end for
function $\mathrm{Generate}(\boldsymbol{W}, c, \xi)$
  $\boldsymbol{p} \leftarrow \boldsymbol{q}(c)/\|\boldsymbol{q}(c)\|_2; \quad \boldsymbol{X}_0 \leftarrow \xi$
  for $k = 0, \dots, K-1$ **do**
    $(\boldsymbol{V}_k, \boldsymbol{U}_k, \boldsymbol{S}_k) \leftarrow \mathrm{Read}(\boldsymbol{W}, c, \boldsymbol{p}, \boldsymbol{X}_k, k)$
    $\boldsymbol{X}_{k+1} \leftarrow \boldsymbol{X}_k + h_k \boldsymbol{V}_k$
  end for
  return $\mathrm{Dec}(\boldsymbol{X}_K)$
end function

Output: trained $\boldsymbol{W}$ and the generator $(c, \xi) \mapsto \mathrm{Generate}(\boldsymbol{W}, c, \xi)$.

trajectories and the clipped objective, with noise coefficient 0.7, KL coefficient 0.01, and clipping parameter 10^{-4} (Appendix C.2). With $\mathcal{T}_i$ the steps of generation i that enter the objective, the update is

$$\nabla^{\mathrm{PG}} = \sum_i \sum_{k \in \mathcal{T}_i} \sum_n \frac{\partial \mathcal{L}_{\mathrm{PG}}}{\partial \boldsymbol{u}_{i,k,n}} \otimes \boldsymbol{p}_i \otimes \boldsymbol{s}_{i,k,n}, \qquad \boldsymbol{W}^+ = \boldsymbol{W} - \eta\gamma\, \nabla^{\mathrm{PG}}. \tag{8}$$

Group normalization and batch reduction belong to $\mathcal{L}_{\mathrm{PG}}$ and enter once; the scalar clipping factor and the detached indices are those of Algorithm 1. SDXL uses the coordinate and density transformation of Appendix C.1, and evaluation calls $\mathrm{Generate}$ with the native schedule of Table 7.

Four forward/update combinations. To separate the two roles of centering, the forward value uses features $\boldsymbol{F}_{\mathrm{f}} \in \{\boldsymbol{F}, \boldsymbol{C}\boldsymbol{F}\}$ and the update uses $\boldsymbol{F}_{\mathrm{u}} \in \{\boldsymbol{F}, \boldsymbol{C}\boldsymbol{F}\}$, implemented by the value-preserving custom derivative

$$\boldsymbol{U} = \mathrm{sg}(\boldsymbol{F}_{\mathrm{f}}\boldsymbol{A}^\top) + \boldsymbol{F}_{\mathrm{u}}\boldsymbol{A}^\top - \mathrm{sg}(\boldsymbol{F}_{\mathrm{u}}\boldsymbol{A}^\top). \tag{9}$$

Its forward value is $\boldsymbol{F}_{\mathrm{f}}\boldsymbol{A}^\top$, and its gradient with respect to $\boldsymbol{W}$ contracts $\boldsymbol{G}^\top \boldsymbol{F}_{\mathrm{u}}$ with $\boldsymbol{p}$. The frozen velocity path is unchanged, and both feature branches are detached in all four combinations. Wrong-index controls use the same mechanism and replace the prompt index or the whole state sequence only in the update, followed by the norm matching of Appendix C.4.

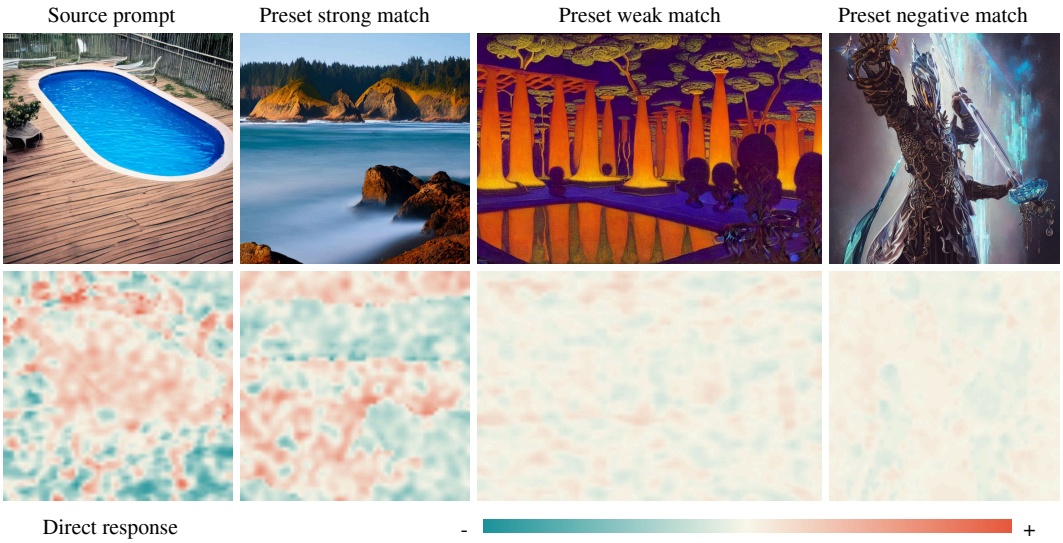


Figure 8: **A source update reaches targets in proportion to prompt similarity and follows their local states.** Top: a source image and three targets whose prompt similarity $\boldsymbol{p}_j^\top \boldsymbol{p}_i$ is set to a strong, a weak, and a negative value. Bottom: $-\eta(\boldsymbol{p}_j^\top \boldsymbol{p}_i)\boldsymbol{S}_j \boldsymbol{S}_i^\top \boldsymbol{G}_i$ at fixed indices, projected onto one signed display coordinate. The prompt factor scales amplitude and sets the sign; the state overlap, together with the source gradient, sets the spatial pattern. Whether the change helps a target also depends on its quality gradient (Eq. (13)).

B Proofs and Analysis of the Update Rule

Section 4 involves three objects. The *update* of $\boldsymbol{W}$ is produced by the backward pass. The *fixed-state response* evaluates the updated tensor at a target's old indices. A *regenerated* target recomputes its indices and its frozen velocity along a changed trajectory. The first two admit exact identities (Sections B.1–B.3); the third needs assumptions on the sampler dynamics and on the quality score (Section B.5). Section B.6 shows why the scalar step and the detached indices are required.

B.1 The gradient of the tensor

Throughout, the prompt vector is nonzero and normalized, all feature maps have fixed parameters, and RMS normalization uses a positive stabilizer, so it is defined for a zero input. The feature is detached before it is used as an index, centering is applied per image and step, $\boldsymbol{W}$ is the only trainable object, and the loss has no other explicit dependence on $\boldsymbol{W}$. A global averaging or clipping factor enters exactly once. These assumptions describe the computation graph actually used for the update, including its truncation or policy-gradient estimator. We index one correction by $a = (i, k, n)$; its entries are $u_{a,\alpha} = \sum_h \sum_b W_{\alpha h b}\, p_{i,h}\, s_{a,b}$.

Lemma 1 (Tensor gradient)**.** *Let $\boldsymbol{g}_a = \partial \mathcal{L}/\partial \boldsymbol{u}_a$ be the total derivative that reverse-mode differentiation assigns to correction a, including all downstream paths retained by the graph. Then*

$$\nabla_{\boldsymbol{W}} \mathcal{L} = \sum_{a=(i,k,n)} \boldsymbol{g}_a \otimes \boldsymbol{p}_i \otimes \boldsymbol{s}_a. \tag{10}$$

The identity holds whether or not the loss separates over sources.

Proof. With the indices of correction a held fixed, $\partial u_{a,\beta}/\partial W_{\alpha h b} = \mathbf{1}\{\alpha = \beta\}\, p_{i,h}\, s_{a,b}$. Multiplying by $g_{a,\beta}$ and summing over all corrections that use $\boldsymbol{W}$ gives Eq. (10). A loss that couples samples changes the values of $\boldsymbol{g}_a$ but leaves this local derivative unchanged; a later correction's index contributes no derivative because of the stop-gradient, while the frozen velocity still passes derivatives through its latent input. □

A *source contribution* is the sum of these terms over one generation, taken from the actual batch loss. For a batch mean over 16 sources, the factor $1/16$ is already inside $\boldsymbol{G}_i$ and is applied once; when the batch gradient is clipped by a scalar γ, every source contribution receives the same γ.

B.2 The fixed-state response and its magnitude

For source i, the rows of $\boldsymbol{S}_i \in \mathbb{R}^{M_i \times d_s}$ and $\boldsymbol{G}_i \in \mathbb{R}^{M_i \times d_v}$ are its state indices and gradients over all M_i locations and steps that receive gradient. A target has $\boldsymbol{S}_j$ and $\boldsymbol{U}_j = \boldsymbol{S}_j \boldsymbol{A}_j^\top$ with $\boldsymbol{A}_j = \boldsymbol{A}(c_j)$; the two row counts may differ.

Proof of Theorem 1. By Lemma 1, the contribution of source i to the update is

$$\Delta \boldsymbol{W}_i = -\eta \sum_{m=1}^{M_i} \boldsymbol{g}_{i,m} \otimes \boldsymbol{p}_i \otimes \boldsymbol{s}_{i,m}, \qquad \Delta \boldsymbol{A}_{j \leftarrow i} = \Delta \boldsymbol{W}_i \times_2 \boldsymbol{p}_j = -\eta \, (\boldsymbol{p}_j^\top \boldsymbol{p}_i) \, \boldsymbol{G}_i^\top \boldsymbol{S}_i. \tag{11}$$

At fixed indices the correction is linear in $\boldsymbol{W}$, so $\Delta \boldsymbol{U}_{j \leftarrow i} = \boldsymbol{S}_j \Delta \boldsymbol{A}_{j \leftarrow i}^\top$, which is Eq. (7). Linearity makes the identity exact for a finite step, and the full update is the sum of its source contributions. □

Proposition 1 (Magnitude of the response)**.** $\|\Delta \boldsymbol{U}_{j \leftarrow i}\|_F \le \eta \, |\boldsymbol{p}_j^\top \boldsymbol{p}_i| \, \|\boldsymbol{S}_j \boldsymbol{S}_i^\top\|_2 \, \|\boldsymbol{G}_i\|_F$*. In particular, the response vanishes when* $\boldsymbol{p}_j^\top \boldsymbol{p}_i = 0$ *or* $\boldsymbol{S}_j \boldsymbol{S}_i^\top = \boldsymbol{0}$*.*

Proof. Apply $\|\boldsymbol{B}\boldsymbol{G}\|_F \le \|\boldsymbol{B}\|_2 \|\boldsymbol{G}\|_F$ to Eq. (7). □

The bound separates prompt similarity, state overlap, and gradient size. Two different requests can still have a nonzero prompt similarity, and visually different states can share feature directions; a zero average of pairwise state similarities also leaves the cross-state matrix nonzero in general, so the diagnostics of Appendix D.4 use the spectral norm of the whole matrix.

The accumulated update also has a closed form. Let training rounds $r = 0, \dots, R-1$ use the source indices $\boldsymbol{p}_{ri}, \boldsymbol{S}_{ri}$, gradients $\boldsymbol{G}_{ri}$, and scalar steps η_r that actually occurred. Summing Eq. (11) over rounds from $\boldsymbol{W} = \boldsymbol{0}$ and contracting with a fixed probe $(\boldsymbol{p}_*, \boldsymbol{S}_*)$ gives

$$\boldsymbol{U}_*^{(R)} = -\sum_{r=0}^{R-1} \eta_r \sum_i (\boldsymbol{p}_*^\top \boldsymbol{p}_{ri}) \, \boldsymbol{S}_* \boldsymbol{S}_{ri}^\top \boldsymbol{G}_{ri}. \tag{12}$$

The trained correction at any prompt and state is thus a sum of past source gradients, each weighted by its prompt similarity and state overlap with the probe.

Figure 8 reads the prompt factor of Eq. (7) on real scenes: the same source update is scaled up, scaled down, or reversed as the prompt similarity changes, while its spatial layout follows each target's own states. Figure 4 isolates the state factor. The source gradient, the target's forward readout, the update norm, and the display scale are fixed, and only the state index under which the update is stored is swapped. The response moves to different regions of the same scene, and the difference map shows that the change is structured along the scene's content, as expected when the state overlap $\boldsymbol{S}_j \boldsymbol{S}_i^\top$ decides where the update acts. Together the two figures show both factors of the product on generated images: the prompt factor controls how strongly and in which direction a target moves, and the state factor controls where.

The swapped index also accounts for the wrong-state results of Table 6. Under a swapped index, the update is stored along state directions of another image, and each target reads it at locations whose states overlap with those directions, which bear no relation to where the source gradient applied. The source itself keeps only the part of its update that lies along its own states: its HPS gain on FLUX falls from 3.23×10^{-2} to 0.65×10^{-2}, and the target change turns from $+4.19 \times 10^{-3}$ to -3.45×10^{-3}. Figure 4 is the spatial counterpart of these numbers. Because the update norm is fixed, the swapped response is as strong as the matched one, but it acts on different content of the scene.

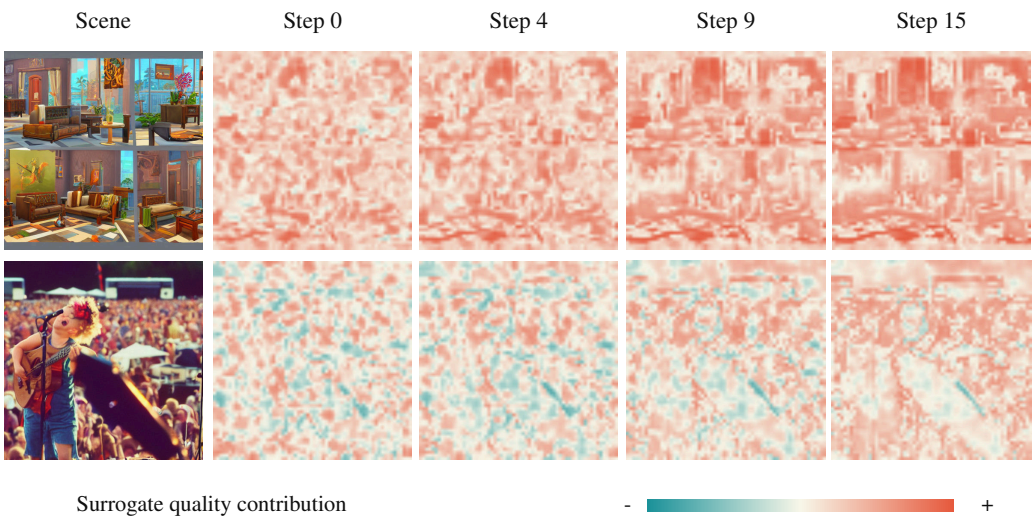


Figure 9: **Local contributions to the quality differential.** Two scenes and their signed per-location contributions at execution steps 0, 4, 9, and 15 on one shared scale, for the quadratic score of Eq. (14). The contributions sum to the directional differential; Figure 25 compares the differential with regenerated ImageReward.

B.3 The first-order change of target quality

To ask whether a response helps a target, consider a differentiable quality score Q_j of the target under additive perturbations of its corrections, with the feature indices clamped. Let $\boldsymbol{H}_j$ be its gradient with respect to the stacked corrections at the old rollout; it can include the frozen sampler's downstream response to a perturbation.

Proposition 2 (Quality differential)**.** *Along the update of source i, the directional derivative of target quality is*

$$D_i Q_j = -\eta\,(\boldsymbol{p}_j^\top \boldsymbol{p}_i)\,\langle \boldsymbol{H}_j, \boldsymbol{S}_j \boldsymbol{S}_i^\top \boldsymbol{G}_i \rangle_F, \qquad |D_i Q_j| \le \eta\,|\boldsymbol{p}_j^\top \boldsymbol{p}_i|\,\|\boldsymbol{S}_j \boldsymbol{S}_i^\top\|_2\,\|\boldsymbol{G}_i\|_F\,\|\boldsymbol{H}_j\|_F. \quad (13)$$

Proof. The differential is the Frobenius inner product of $\boldsymbol{H}_j$ with the response in Eq. (7); Cauchy–Schwarz and Proposition 1 give the bound. □

A positive prompt similarity and a nonzero state overlap leave the sign open: for $\boldsymbol{B} = \boldsymbol{S}_j \boldsymbol{S}_i^\top \boldsymbol{G}_i \neq \boldsymbol{0}$, the gradients $\boldsymbol{H}_j = \boldsymbol{B}$ and $\boldsymbol{H}_j = -\boldsymbol{B}$ give opposite signs. The target's own quality gradient therefore decides whether a response helps.

Local contributions. Figure 9 displays where the differential comes from, using the quadratic score $Q(\boldsymbol{U}) = -\|\boldsymbol{U} - \boldsymbol{R}\|_F^2/(2N)$ for a fixed reference $\boldsymbol{R}$. Its gradient is $\boldsymbol{H} = (\boldsymbol{R} - \boldsymbol{U})/N$, location n contributes $\sum_a H_{n,a} \Delta U^{\mathrm{dir}}_{n,a}$, and at fixed inputs the finite change is exactly

$$Q(\boldsymbol{U} + \Delta\boldsymbol{U}^{\mathrm{dir}}) - Q(\boldsymbol{U}) = \langle \boldsymbol{H}, \Delta\boldsymbol{U}^{\mathrm{dir}} \rangle_F - \frac{\|\Delta\boldsymbol{U}^{\mathrm{dir}}\|_F^2}{2N}. \quad (14)$$

The signed fields show which locations support or oppose the quality direction, and the quadratic term is the finite-step correction. Positive and negative regions shift across steps, so a large state overlap and a favorable quality direction are separate conditions.

B.4 Removal of the image-wide term

At one image and step, $\boldsymbol{F} = \boldsymbol{1}\bar{\boldsymbol{f}}^\top + \boldsymbol{C}\boldsymbol{F}$ and $\boldsymbol{G} = \boldsymbol{1}\bar{\boldsymbol{g}}^\top + \boldsymbol{C}\boldsymbol{G}$ with $\boldsymbol{1}^\top \boldsymbol{C}\boldsymbol{F} = \boldsymbol{0}$ and $\boldsymbol{1}^\top \boldsymbol{C}\boldsymbol{G} = \boldsymbol{0}$. The argument holds at every image and step before the contributions are summed into $\boldsymbol{W}$.

Proof of Theorem 2. Expanding the product of the two decompositions, both cross terms vanish because the centered matrices have zero column sums, which gives Eq. (3). Since $\boldsymbol{C}\boldsymbol{1} = \boldsymbol{0}$ and

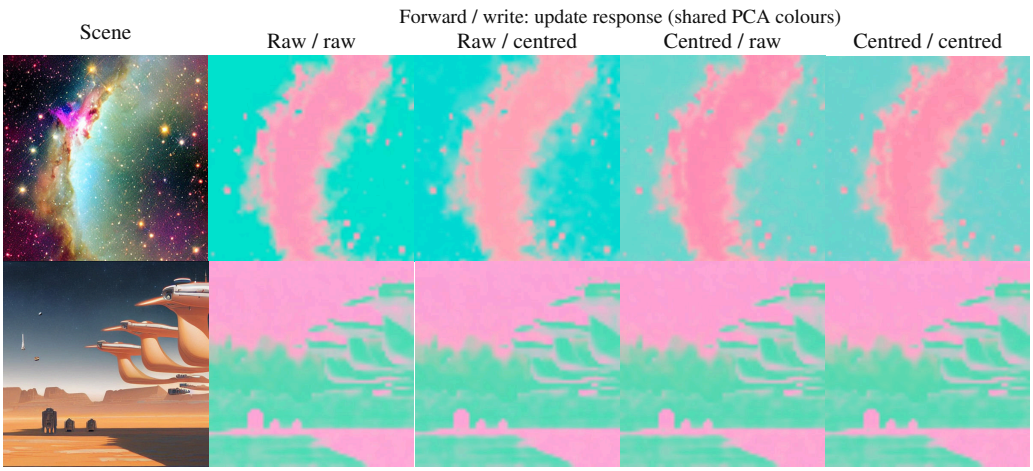


Figure 10: **One update under the four forward/update combinations.** Each scene is followed by the change of its correction under F/F, F/CF, CF/F, and CF/CF in shared PCA colors. Within a forward branch, the two updates share the source gradient and its clipping factor, and the zero-mean term keeps its scale; the two forward branches use their own gradients.

$\boldsymbol{C}^\top = \boldsymbol{C} = \boldsymbol{C}^2$, $\boldsymbol{G}^\top\boldsymbol{CF} = (\boldsymbol{CG})^\top(\boldsymbol{CF})$. A constant gradient $\boldsymbol{G} = \mathbf{1}\boldsymbol{a}^\top$ gives $\boldsymbol{G}^\top\boldsymbol{CF} = \boldsymbol{a}\mathbf{1}^\top\boldsymbol{CF} = \mathbf{0}$. A zero-mean gradient, $\mathbf{1}^\top\boldsymbol{G} = \mathbf{0}$, gives $\boldsymbol{G}^\top\boldsymbol{CF} = \boldsymbol{G}^\top\boldsymbol{F} - \boldsymbol{G}^\top\mathbf{1}\bar{\boldsymbol{f}}^\top = \boldsymbol{G}^\top\boldsymbol{F}$. The common prompt factor preserves both statements for the tensor update. □

Proposition 3 (Uniqueness)**.** *Among fixed matrices $\boldsymbol{P} \in \mathbb{R}^{N\times N}$ applied as $\boldsymbol{PF}$, $\boldsymbol{C}$ is the only one such that, for all $\boldsymbol{F}$ and $\boldsymbol{G}$, a constant $\boldsymbol{G}$ gives $\boldsymbol{G}^\top\boldsymbol{PF} = \mathbf{0}$ and a zero-mean $\boldsymbol{G}$ gives $\boldsymbol{G}^\top\boldsymbol{PF} = \boldsymbol{G}^\top\boldsymbol{F}$.*

Proof. Constant gradients $\boldsymbol{G} = \mathbf{1}\boldsymbol{a}^\top$ with arbitrary $\boldsymbol{a}$ and $\boldsymbol{F}$ force $\mathbf{1}^\top\boldsymbol{P} = \mathbf{0}^\top$. For any $\boldsymbol{q} \perp \mathbf{1}$, a gradient column equal to $\boldsymbol{q}$ and arbitrary $\boldsymbol{F}$ force $\boldsymbol{q}^\top(\boldsymbol{P} - \boldsymbol{I}) = \mathbf{0}^\top$, so every column of $\boldsymbol{P} - \boldsymbol{I}$ lies in the span of $\mathbf{1}$ and $\boldsymbol{P} = \boldsymbol{I} + \mathbf{1}\boldsymbol{b}^\top$. The first condition then gives $\mathbf{1}^\top + N\boldsymbol{b}^\top = \mathbf{0}^\top$, hence $\boldsymbol{P} = \boldsymbol{I} - \mathbf{1}\mathbf{1}^\top/N = \boldsymbol{C}$; the proof of Theorem 2 shows that $\boldsymbol{C}$ has both properties. □

The removed term $N\bar{\boldsymbol{g}}\bar{\boldsymbol{f}}^\top$ depends on the resolution through N, $\bar{\boldsymbol{g}}$, and $\bar{\boldsymbol{f}}$. Forward centering gives $\mathbf{1}^\top\boldsymbol{U} = \mathbf{1}^\top\boldsymbol{CFA}^\top = \mathbf{0}$: for $N = 1$ the adapter vanishes, and for $N > 1$ it keeps all zero-mean velocity corrections, whose effect on the decoded image need not have zero mean after the nonlinear sampler and decoder. Figure 10 shows one update of a trained model under the four forward/update combinations.

B.5 Recomputed indices and the sampler trajectory

For a target regenerated from the same initial noise, adding and subtracting $\boldsymbol{S}_j(\boldsymbol{A}_j^+)^\top$ gives

$$\begin{aligned} \boldsymbol{U}_j^+ - \boldsymbol{U}_j &= \boldsymbol{S}_j(\boldsymbol{A}_j^+ - \boldsymbol{A}_j)^\top + (\boldsymbol{S}_j^+ - \boldsymbol{S}_j)(\boldsymbol{A}_j^+)^\top, \\ \left\|\boldsymbol{U}_j^+ - \boldsymbol{U}_j - \Delta\boldsymbol{U}_j^{\mathrm{dir}}\right\|_F &\le \|\boldsymbol{S}_j^+ - \boldsymbol{S}_j\|_F\,\|\boldsymbol{A}_j^+\|_2, \end{aligned} \tag{15}$$

where $\Delta\boldsymbol{U}_j^{\mathrm{dir}} = \boldsymbol{S}_j(\boldsymbol{A}_j^+ - \boldsymbol{A}_j)^\top$ is the fixed-state response of Eq. (7). A cached-index intervention sets the target's indices to their old values, while the latent and the frozen velocity still follow the new trajectory; the paired endpoint difference therefore measures the effect of recomputing the indices at a given checkpoint.

For the trajectory, index Euler steps in execution order with signed step sizes h_k, $\boldsymbol{X}_{k+1} = \boldsymbol{X}_k + h_k\{\boldsymbol{V}_k^0(\boldsymbol{X}_k) + \boldsymbol{U}_k(\boldsymbol{W}, \boldsymbol{X}_k)\}$. The updated trajectory uses $\boldsymbol{W}^+$ and starts at $\boldsymbol{X}_0^+ = \boldsymbol{X}_0$. Subtracting and telescoping gives the endpoint difference

$$\boldsymbol{X}_K^+ - \boldsymbol{X}_K = \sum_{k=0}^{K-1} h_k\{\boldsymbol{V}_k^0(\boldsymbol{X}_k^+) - \boldsymbol{V}_k^0(\boldsymbol{X}_k) + \boldsymbol{U}_k^+ - \boldsymbol{U}_k\}, \tag{16}$$

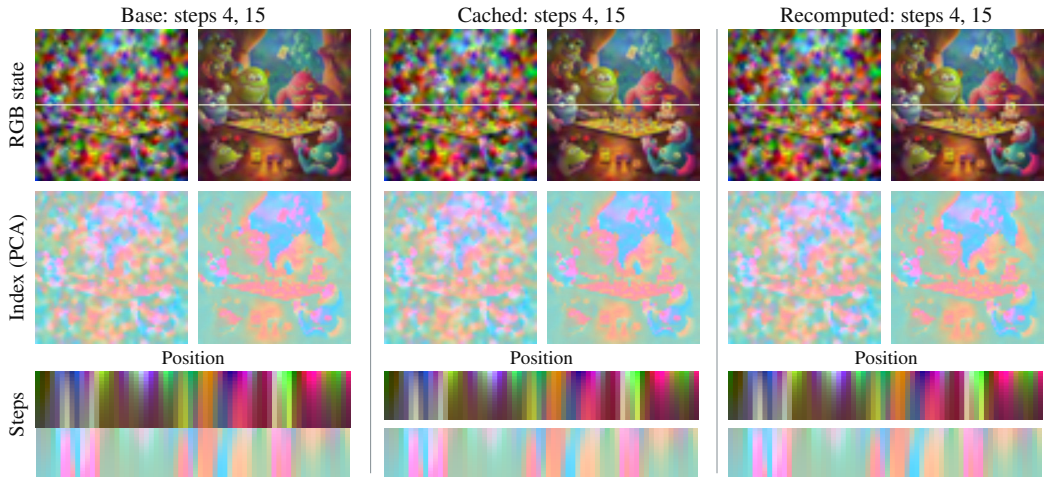


Figure 11: **States and indices under cached and recomputed indices.** Base, cached-index, and recomputed-index trajectories at execution steps 4 and 15. Top rows: fixed RGB readout of the state; middle rows: the index in a shared PCA basis; bottom strips: one horizontal slice at normalized $y = 0.5$ across execution steps. Cached indices equal the old indices by construction while the latent and frozen velocity evolve, which separates the second term of Equation 15.

whose first term is the frozen model's response to the changed trajectory; the decoder then maps the latent endpoint to the image.

Proposition 4 (Trajectory bound)**.** *Suppose that, on the segments joining the paired latents at step k, the frozen velocity is $L_{v,k}$-Lipschitz and the state map is $L_{S,k}$-Lipschitz in Frobenius norm. With $d_k = \|\boldsymbol{S}_k(\boldsymbol{X}_k)(\boldsymbol{A}^+ - \boldsymbol{A})^\top\|_F$ and $L_k = L_{v,k} + L_{S,k}\|\boldsymbol{A}^+\|_2$,*

$$\|\boldsymbol{X}_K^+ - \boldsymbol{X}_K\|_F \le \sum_{k=0}^{K-1} |h_k|\, d_k \prod_{\ell=k+1}^{K-1} \big(1 + |h_\ell| L_\ell\big). \tag{17}$$

Proof. Let $e_k = \|\boldsymbol{X}_k^+ - \boldsymbol{X}_k\|_F$. By Eq. (15), the correction difference splits into the old-state term, bounded by d_k, and the new operator acting on the state difference, bounded by $L_{S,k}\|\boldsymbol{A}^+\|_2 e_k$. With the velocity bound, $e_{k+1} \le (1 + |h_k|L_k)e_k + |h_k|d_k$; unrolling from $e_0 = 0$ gives the result, with an empty product equal to one. □

The bound shows how a small update at old states can have a larger endpoint effect when later steps amplify perturbations. If the update at old states is exactly zero at every step and sampling is deterministic, the two trajectories coincide by induction; a zero overlap at one state or step is not enough for this conclusion. A bound on the decoded image or its quality needs matching regularity assumptions on the decoder and the score. Figure 11 shows the cached and recomputed indices along a trajectory.

B.6 Why the scalar step and detached indices are required

Coordinate-wise scaling can create a response where the prompt factor is zero. Take one velocity channel, one state channel, two prompt channels, centered states $\boldsymbol{S} = (1, -1)^\top$, gradients $\boldsymbol{G} = (1, -1)^\top$, and prompts $\boldsymbol{p} = (1, 1)^\top/\sqrt{2}$ and $\boldsymbol{p}' = (1, -1)^\top/\sqrt{2}$. The source gradient in prompt coordinates is $2\boldsymbol{p}$, and scalar SGD gives a zero target contraction because $\boldsymbol{p}'^\top\boldsymbol{p} = 0$. Scaling the two coordinates by $\boldsymbol{D} = \mathrm{diag}(1, 2)$ instead gives

$$\boldsymbol{p}'^\top(-2\eta\, \boldsymbol{D}\boldsymbol{p}) = \eta \neq 0. \tag{18}$$

Rescaling the whole update afterwards multiplies this value by a scalar and keeps it nonzero, so coordinate scaling changes the response even at matched update norm.

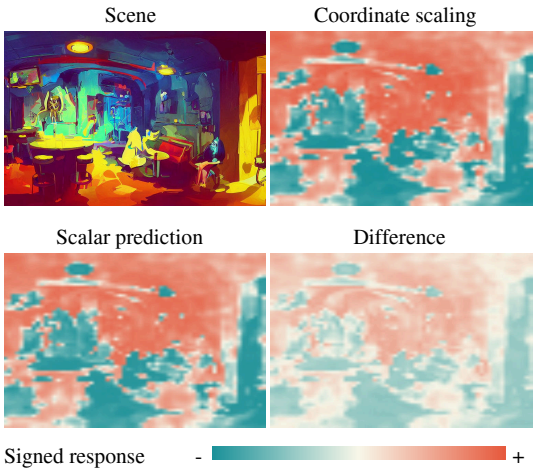


Figure 12: **Coordinate scaling changes the response at equal norm.** Scene, response after a diagonal (RMSProp-like) step, the scalar-SGD prediction of Equation 7, and their difference, in row-major order. Indices, source gradient, update norm, and signed display scale are fixed.

Figure 12 extends the example to a spatial response. Its bottom-right field compares the coordinate-scaled response with the scalar prediction on the same display scale, and the difference persists after the total update size is matched. A uniform scalar step multiplies every response entry by the same number and keeps the factorization; a diagonal step changes the direction of the update in parameter coordinates and removes the cancellations between prompt coordinates that give a zero response.

The other conditions enter the same way. Training the backbone or the feature encoder adds parameter paths whose Jacobians join the cross-Jacobian; removing the stop-gradient adds index derivatives to Eq. (10); a residual bias adds an update path without the centered state factor; momentum mixes earlier gradients into the current source decomposition; and weight decay adds a direct parameter term. A general hypernetwork has its own Jacobian and can express interactions beyond this tensor, but it does not enforce the factorization. Eq. (7) therefore describes the tensor readout with detached indices and scalar updates defined in Section 4, and these are the settings used in all experiments.

C Implementation Details

C.1 Native interfaces and tensor sizes

Table 7: Native schedules and adapter sizes for FLUX, SD3.5, and SDXL (Black Forest Labs, 2024; Stability AI, 2024; Podell et al., 2024). Shape: $d_v \times d_c \times d_s$. FLUX uses guidance embeddings; the others use classifier-free scales.

Model	Tensor shape	Parameters	Steps	Guide
FLUX.1-dev	$64 \times 768 \times 128$	6,291,456	50	3.5
SD3.5-Medium	$16 \times 2048 \times 32$	1,048,576	28	4.5
SDXL-base 1.0	$4 \times 1280 \times 8$	40,960	50	5.0

All outputs have resolution 1024×1024, and each model keeps its step count and guidance interface (Table 7). FLUX.1-dev uses packed latents and guidance embeddings, SD3.5-Medium uses the flow-Euler sampler, and SDXL uses the sigma-Euler sampler without a refiner. The prompt index is the pooled vector each model already computes: the pooled CLIP vector of FLUX, the concatenated pooled vector of SD3.5, and the pooled second-text-encoder vector of SDXL; the denoiser keeps its token-level conditioning. Text limits are 512 tokens for FLUX, 256 for SD3.5, and 77 CLIP tokens for SDXL.

The state vector concatenates the native latent and the frozen velocity after guidance. For $\boldsymbol{z} \in \mathbb{R}^{d_s}$, normalization is $\boldsymbol{z}/\sqrt{\|\boldsymbol{z}\|_2^2/d_s + \epsilon}$ with the FP32 machine epsilon $\epsilon = 2^{-23}$ and no affine parameters, and the spatial mean is subtracted after normalization. $\boldsymbol{W}$ is stored and updated in FP32, including contractions and reduction accumulators, while the frozen backbone runs in BF16; the correction is cast at the native sampler interface, and the number of denoiser calls is unchanged.

SDXL coordinates. Let $\boldsymbol{y}$ be the native scheduler latent and σ the noise level. The U-Net may receive a scaled input, but the state index uses $[\boldsymbol{y}; \boldsymbol{\epsilon}_0(\boldsymbol{y}, \sigma, c)]$, where $\boldsymbol{\epsilon}_0$ is the frozen guided noise prediction and also the sigma-Euler velocity. With zero churn and an empty negative prompt, the actual update is

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k + (\sigma_{k+1} - \sigma_k)\{\boldsymbol{\epsilon}_0(\boldsymbol{y}_k, \sigma_k, c) + \boldsymbol{u}_k\}. \tag{19}$$

Direct training uses this interface. For policy objectives that need a flow coordinate, we use

$$t = \frac{\sigma}{1+\sigma}, \qquad \boldsymbol{z} = \frac{\boldsymbol{y}}{1+\sigma}, \qquad \boldsymbol{v}_{\boldsymbol{z}} = (1+\sigma)\boldsymbol{\epsilon}_0 - \boldsymbol{y}, \tag{20}$$

in which the adapter contributes $(1+\sigma)\boldsymbol{u}$ while its index remains the native $[\boldsymbol{y}; \boldsymbol{\epsilon}_0]$. Differentiating $\boldsymbol{z}$ with respect to t verifies the velocity; on the mapped grid, $\Delta t = \Delta\sigma/[(1+\sigma_k)(1+\sigma_{k+1})]$, and an Euler step in $\boldsymbol{z}$ maps exactly to Eq. (19). Stochastic training uses the transformed transition density: for $\boldsymbol{y}_{k+1} = (1+\sigma_{k+1})\boldsymbol{z}_{k+1}$ in d dimensions, $\log p_{\boldsymbol{y}} = \log p_{\boldsymbol{z}} - d\log(1+\sigma_{k+1})$. The likelihood uses the density of the selected coordinate, and evaluation uses the native scheduler.

C.2 Prompt splits, learning-rate selection, and matched training

We use the DiffusionDB 2M prompts without their images. Normalization applies Unicode NFKC, whitespace consolidation, and case folding. Exact matches and character five-gram Jaccard matches at threshold 0.8 to any test prompt are removed from the learning pool; within the pool, exact and near duplicates are merged before a deterministic split into 8,192 training, 512 validation, and 512 calibration prompts. Test lists stay complete, calibration supplies no preference labels, generation seeds are paired within comparisons, and the training seeds are 17, 29, and 43.

Direct training minimizes the negative raw HPSv2.1 similarity with the full forward trajectory and backpropagation through the final sampling step; tables multiply the raw similarity by 100. PSTune and its tensor controls use SGD with zero momentum, zero weight decay, and global gradient clipping at norm one, with learning-rate candidates 10^{-4}, 10^{-3}, and 10^{-2}. The shared LoRA has rank and scaling 32 on the attention and feed-forward projections and uses AdamW with coefficients $(0.9, 0.999)$, stabilizer 10^{-8}, zero weight decay, and candidates 6×10^{-5}, 2×10^{-4}, and 6×10^{-4}. It has 130,744,320 trainable parameters on FLUX, 46,989,312 on SD3.5, and 83,722,240 on SDXL.

Each candidate trains for 256 updates with selection seed 17, and validation ImageReward picks the learning rate, with ties resolved toward the smaller value. The three final runs restart from zero initialization and stop at 1,024 updates. The matched policy-gradient condition keeps the tensor or LoRA interface and its optimizer and replaces the objective with Flow-GRPO: groups of 24, ten-step stochastic trajectories, noise coefficient 0.7, KL coefficient 0.01, and clipping parameter 10^{-4}, again with three candidates of 256 updates each. This condition changes the objective at a fixed budget and is separate from the external Flow-GRPO run with its native schedule.

C.3 External methods

Table 8: FLUX resource counts in thousands. Training includes RSA-FT replay and the D2-Align clean cache. Selection covers three learning-rate trials; validation and final tests are counted separately. Method citations appear in Table 1.

Method	Training (k)		Selection
	Images	Rewards	Images (k)
PSTune, direct	16.4	16.4	12.3
LoRA, direct (Hu et al., 2022)	16.4	16.4	12.3
PSTune, PG	24.6	24.6	18.4
Flow-GRPO (Liu et al., 2025)	2304.0	2304.0	6912.0
GRPO-Guard (Wang et al., 2026)	587.5	587.5	1762.6
DiverseGRPO (Liu et al., 2026)	384.0	384.0	1152.0
DiffusionNFT (Zheng et al., 2026)	2304.0	2304.0	6912.0
RSA-FT (Kim & Sim, 2026)	16.0	24.0	48.0
D2-Align (Chen et al., 2026)	8.2	8.2	24.7
DGPO (Luo et al., 2026)	48.0	48.0	144.0

The seven external methods (at native guidance on FLUX; on SD3.5 and SDXL, DiffusionNFT runs at guidance one) optimize the same HPS reward on the same training prompts with their native objectives and schedules: Flow-GRPO (Liu et al., 2025) and DiverseGRPO (Liu et al., 2026) with 4,000 optimizer updates, GRPO-Guard (Wang et al., 2026) with 1,020, DiffusionNFT (Zheng et al., 2026) with 2,000 collection epochs, RSA-FT (Kim & Sim, 2026) with 2,000 ReFL updates including its parameter-perturbation replay, D2-Align (Chen et al., 2026) with a 3,000-update direction-learning stage and its clean-image cache followed by 20 policy updates, and DGPO (Luo et al., 2026) with 2,000 groups of 24. The methods differ in how they form the gradient: Flow-GRPO uses group-relative policy gradients, DiverseGRPO adds a diversity-aware advantage, GRPO-Guard regulates the clipped ratio, DiffusionNFT trains on the forward process, RSA-FT backpropagates the reward and penalizes its sharpness, D2-Align decouples a learned preference direction, and DGPO learns from group preferences. Each external method receives three learning-rate candidates evaluated over its native schedule. Training uses four A100 80GB PCIe GPUs.

Training budgets. Table 8 counts the images generated and the reward calls made by these schedules on FLUX. The budgets span more than two orders of magnitude. Flow-GRPO and DiffusionNFT generate 2,304k training images each, GRPO-Guard 587.5k, and DiverseGRPO 384k, whereas direct PSTune and LoRA generate 16.4k and PG PSTune 24.6k. RSA-FT generates 16.0k images but makes 24.0k reward calls because of its replay, and D2-Align uses 8.2k.

The budgets rule out a larger training set as the source of PSTune's advantage. Against the methods with smaller or similar budgets, D2-Align and RSA-FT, PSTune reaches a higher FLUX GenEval ImageReward (1.201 against 1.078 and 1.050). Against the methods with 140 times its budget, PSTune still has the highest ImageReward, while Flow-GRPO reaches a higher HPS (Table 1). The selection budget follows the same pattern: 12.3k images for PSTune against 6,912k for Flow-GRPO and DiffusionNFT, because the three learning-rate trials of each method repeat its native schedule. Matched LoRA uses exactly PSTune's budget, which makes it the controlled comparison for the parameterization.

Guidance. DiffusionNFT's SD3.5 and SDXL settings use guidance one, so we train separate guidance-one PSTune and LoRA models with the corresponding data, budgets, and selection rule (Table 10). Within a backbone and guidance setting, all test outputs share the resolution, prompt list, generation seeds, and native sampler.

C.4 Controlled updates

Wrong indices and RMSProp. A wrong-index control takes the prompt vector, or the whole state sequence, of another sample through a nonzero cyclic shift of the batch. Each arm computes its gradient on its own rollout, rescales the whole wrong-index tensor gradient by a positive scalar to the norm of that rollout's correct-index gradient, and then applies the common clipping rule. RMSProp uses decay 0.99, stabilizer 10^{-8}, zero momentum, an uncentered second moment, and no weight decay, followed by matching to the current SGD update norm.

Single-source interventions. These use checkpoints 0, 512, and 1,024 of the same three training runs. Each seed contributes 32 batches of 16 training prompts, with one source per batch chosen deterministically before scoring. Target prompts are assigned to sources deterministically, and all target images are kept. Each source update starts from the same old checkpoint with its original batch-mean contribution and the full-batch clipping factor. The four arms are correct indices, a wrong prompt, a wrong whole-image state, and correct indices with cached target indices. At check-

point 0, a shared-LoRA source control decomposes its update with the actual full-batch AdamW preconditioner, which compares the complete LoRA and tensor updates at initialization.

Removing the image-wide term. This intervention starts from each PSTune checkpoint at update 1,023 and its fixed final source batch. Each forward choice computes its own rollout and $\boldsymbol{G}$; within that choice, the raw and centered updates share $(\boldsymbol{G}, \boldsymbol{p}, \boldsymbol{F})$ and the learning rate. Their common clipping factor comes from the forward branch's native batch update, $\boldsymbol{G}^\top \boldsymbol{F}$ for raw forward features and $\boldsymbol{G}^\top \boldsymbol{C}\boldsymbol{F}$ for centered ones, and no norm adjustment follows the removal. Targets and the 16 sources are regenerated from paired initial noise. The centered-forward, centered-update endpoint therefore equals the PSTune checkpoint at update 1,024. The full-training comparisons of the four combinations instead measure the whole effect of a forward choice, including its changed states and gradients.

D Evaluation Protocols

D.1 Benchmarks and metrics

HPS and GenEval. The HPS benchmark uses all 3,200 prompts, 800 in each of four styles, with one image per prompt, and its overall score weights the four styles equally (Wu et al., 2023). GenEval uses all 553 prompts with four images each (Ghosh et al., 2023); its six tasks have 80, 99, 80, 94, 100, and 100 prompts for single objects, two objects, counting, colors, position, and color attribution. Each task aggregates its own correctness counts, and the overall score averages the six task rates.

T2I-CompBench and DPG-Bench. T2I-CompBench uses its six original test categories with 300 prompts and ten images per prompt (Huang et al., 2023): BLIP-VQA scores color, shape, and texture binding, UniDet scores spatial relations, the official CLIP procedure scores non-spatial relations, and the official three-in-one evaluator scores complex composition. DPG-Bench uses all 1,065 dense prompts with four images each and its dependency-aware question evaluation (Hu et al., 2024). Our loader keeps the first real question, which the inherited header-skipping condition would drop, so all 14,392 questions are evaluated with the original ordered, in-place dependency gating. Category scores average raw answers, and the overall score uses dependency-gated answers.

GenAI-Bench and PartiPrompts. GenAI-Bench uses the full 1,600-prompt image list (Li et al., 2024) with CLIP-FlanT5-XXL VQAScore and the legacy interface of `t2v_metrics` 3.0 (Lin et al., 2024). The full-list score averages prompts directly; the legacy union of skill-tagged prompts has 1,593 prompts, because seven prompts carry no tag and the 12 skill groups overlap. PartiPrompts uses all 1,632 rows of the public prompt repository with one output per prompt (Yu et al., 2022); we report VQAScore and the human judgments of Appendix D.2. For FLUX and SD3.5, matched comparisons cover all six lists and external comparisons the five lists of Table 1; for SDXL, both cover the HPS, GenEval, and T2I-CompBench lists. The mechanism controls use full GenEval on all three models, and PartiPrompts where Tables 4–5 report it.

Scores. ImageReward-v1.0 is a scalar score (Xu et al., 2023). PickScore is the raw normalized image–text inner product multiplied by its learned logit scale (Kirstain et al., 2023). The supplementary aesthetic score uses the LAION CLIP ViT-L/14 MLP trained on SAC, Logos, and AVA ratings. HPS is the only training reward, ImageReward is used for validation selection, and PickScore, the aesthetic score, and human judgments enter neither training nor selection. LPIPS is version 0.1 with AlexNet (Zhang et al., 2018), computed on all within-prompt pairs and aggregated with each benchmark's task weights: six pairs per GenEval prompt and 45 per T2I-CompBench prompt.

Visualizing within-prompt variation. LPIPS summarizes how much the images of one prompt differ. Figure 5 in Section 5.1 shows the same quantity spatially for four archived Stable Diffusion outputs of one DiffusionDB prompt (guidance 7, `k_lms` sampler, 50 steps, seeds 3918620792, 2575836988, 1996930743, and 3566989601). After sRGB D65-to-CIELAB conversion, each pixel shows $\sqrt{\operatorname{tr} \widehat{\operatorname{Cov}}(L, a, b)}$ over the four samples with the unbiased covariance. The coordinates are not registered, so the map includes both color changes and object displacement; the adapter comparisons use LPIPS on model outputs.

D.2 HUMAN PREFERENCE JUDGMENTS

All 1,632 PartiPrompts are judged for each method pair and training seed, with one image per prompt and method. The two images of a pair share the prompt and initial noise and are shown without labels in random left–right order. Five independent judgments per pair answer three questions: image–text match, realism, and overall visual preference; ties are allowed. This gives 8,160 judgments per question and seed, 24,480 over three seeds. For cross-prompt diversity, eight consecutive prompts of the list form a page; the 204 pages per pair and seed each receive five judgments (1,020 per seed, 3,060 in total), and raters compare the variety of expression while considering the requested content and style. This page-level question complements within-prompt LPIPS.

For each question and seed, with n_{win}, n_{tie}, and n_{loss} wins, ties, and losses, the net preference is $100(n_{\mathrm{win}} - n_{\mathrm{loss}})/(n_{\mathrm{win}} + n_{\mathrm{tie}} + n_{\mathrm{loss}})$, with ties in the denominator. The four questions are aggregated separately, and tables report the mean of the three seed-level values. The training seed is the experimental replicate; image pairs and pages are the judgment units within it.

D.3 SEED UNCERTAINTY AND PAIRED CONTRASTS

For three seed aggregates z_1, z_2, z_3, the reported interval is

$$\bar{z} \pm t_{0.975,2}\,\frac{s_z}{\sqrt{3}}, \qquad s_z^2 = \frac{1}{2}\sum_{r=1}^{3}(z_r - \bar{z})^2, \qquad t_{0.975,2} \simeq 4.303. \tag{21}$$

A paired comparison first forms $z_r = a_r - b_r$ within each seed, so the interval keeps the covariance between methods; intervals are not clipped to metric ranges. The three groups of the frozen base vary generation randomness only. The single-source analysis keeps three intervention seed groups whose 32 sources are averaged under a fixed target assignment, and the checkpoints share their training runs. The analysis of the final batch uses three source batches, one per seed. Tables print seed means; standard deviations and intervals are used in the analysis and figures.

D.4 MECHANISM DIAGNOSTICS AND DISPLAY CONVENTIONS

The diagnostics use the raw prompt similarity $\boldsymbol{p}_j^\top \boldsymbol{p}_i$, the cross-state spectral norm $\|\boldsymbol{S}_j \boldsymbol{S}_i^\top\|_2$, and the quality differential of Eq. (13). The first keeps the sign of the prompt factor, the second measures the largest amplification by the state factor, and together with the step size and gradient norm they give the bound of Proposition 1; the third adds the target's quality direction and can take either sign, so two targets with similar response magnitudes can have opposite quality changes. For efficiency, thin QR factorizations $\boldsymbol{S}_j = \boldsymbol{Q}_j \boldsymbol{T}_j$ and $\boldsymbol{S}_i = \boldsymbol{Q}_i \boldsymbol{T}_i$ give $\|\boldsymbol{S}_j \boldsymbol{S}_i^\top\|_2 = \|\boldsymbol{T}_j \boldsymbol{T}_i^\top\|_2$ without forming the spatial cross-Gram matrix.

Each checkpoint's diagnostics use its old rollout, where the response is exact and the differential is its first-order effect. We compare the differential with fully regenerated ImageReward changes at updates 0, 512, and 1,024; Appendix F.4 defines the source assignment, error, and sign agreement.

Display conventions. Signed fields in Figures 4, 12, and 9 use one projection and the transform $\operatorname{asinh}(v/(m/12))/\operatorname{asinh}(12)$, where m is the largest absolute projected value across the compared panels; this keeps the sign and shows weak structure on one scale per comparison. PCA fields in Figures 11 and 10 use a shared basis and range within each comparison. Scene references in the spatial displays come from DiffusionDB, and outlines mark spatial correspondence.

E ADDITIONAL RESULTS ON QUALITY, ADHERENCE, AND VARIETY

E.1 TASK PROFILES

Figure 13 resolves the aggregate FLUX results by skill. PSTune stays close to the frozen base on GenEval, T2I-CompBench, and GenAI-Bench, whereas shared LoRA loses accuracy on most skills. PSTune remains below the base on GenEval counting and on T2I-CompBench shape binding, and DGPO exceeds PSTune on several skills; the aggregate retention result therefore holds with these skill-level exceptions.

Table 9: T2I-CompBench scores (×100; three-seed means). Each backbone contains the frozen base and matched direct/PG LoRA–PSTune pairs. Bold marks the largest value within each backbone and task.

Method	Attribute binding			Relationships		Composition
	Color	Shape	Texture	Spatial	Non-spatial	Complex
FLUX.1-dev (12B) — T2I-CompBench						
Frozen base	74.1	**57.9**	69.3	28.4	31.1	39.9
LoRA, direct (Hu et al., 2022)	68.2	53.9	62.7	26.9	30.6	36.4
PSTune, direct	74.3	56.6	**69.7**	29.0	31.3	**40.1**
LoRA, PG (Hu et al., 2022)	69.6	54.6	64.9	28.0	31.0	37.8
PSTune, PG	**74.4**	57.2	68.8	**29.1**	**31.5**	40.0
SD3.5-Medium (2.5B) — T2I-CompBench						
Frozen base	80.1	56.9	73.2	28.3	31.6	38.9
LoRA, direct (Hu et al., 2022)	71.5	52.8	66.2	26.4	30.6	34.3
PSTune, direct	**80.4**	56.5	**73.8**	**28.6**	**32.2**	**39.2**
LoRA, PG (Hu et al., 2022)	74.5	54.1	68.8	27.2	30.9	36.1
PSTune, PG	80.1	**57.4**	73.4	28.5	31.4	38.8
SDXL-base 1.0 (2.6B) — T2I-CompBench						
Frozen base	59.1	46.3	54.0	19.0	30.1	32.1
LoRA, direct (Hu et al., 2022)	53.5	42.6	48.0	17.2	28.3	28.4
PSTune, direct	59.1	**47.1**	54.1	19.2	29.7	32.4
LoRA, PG (Hu et al., 2022)	55.3	44.4	50.6	18.6	29.5	30.1
PSTune, PG	**59.3**	46.4	**54.2**	**19.6**	**30.2**	**32.8**

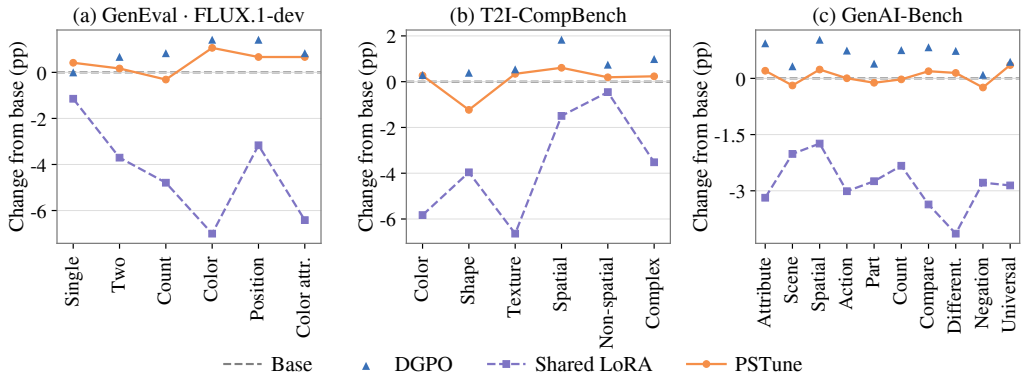


Figure 13: **Skill-level changes behind the aggregate scores.** FLUX, direct training at update 1,024 with native guidance. Paired differences from the frozen base, averaged over three seeds and multiplied by 100: GenEval (553 prompts, four images each), T2I-CompBench (300 prompts per task, ten images each), and GenAI-Bench (1,600 prompts, one image each). The ten GenAI skill tags overlap and are evaluated separately.

Table 9 lists the absolute T2I-CompBench scores. Every PSTune row exceeds its matched LoRA row in all six categories; for FLUX direct training the gains are 6.1 points in color binding, 7.0 in texture, and 2.1 in spatial relations. The comparison with the frozen base locates where LoRA loses content. Its largest drops are in attribute binding, up to 8.6 points in SD3.5 color and 7.0 in SD3.5 texture, while spatial and non-spatial relations lose at most 1.9 points. These categories name a specific color, shape, or material, which is the content a preferred appearance most directly overwrites when it spreads to all prompts. PSTune stays within 1.3 points of the base in all 36 PSTune cells of the table, with the largest drop in FLUX shape binding. Figure 14 shows the same categories on single images.

E.2 Within-prompt variation

LPIPS measures how different the images of one prompt are. Figure 15 shows five LoRA and five PSTune outputs for two prompts: read vertically for the requested scene and style, and horizontally for changes in viewpoint, placement, and composition. Figure 16 repeats the comparison for a silver

robot among colorful eggs and a ginger cat playing checkers. In both, PSTune varies composition while keeping the request, which matches its near-base LPIPS in Tables 1 and 2.

The LPIPS numbers show the same pattern on every backbone. Direct LoRA lowers within-prompt LPIPS below the frozen base by 7.51 points on FLUX, 8.62 on SD3.5, and 7.89 on SDXL, and PG LoRA by 5.88, 6.69, and 5.71. PSTune stays within 0.3 points of the base in all six settings (Tables 1 and 2). The prompt–state rule explains the difference. A shared update applies the same weight change to every sample of a prompt and pushes all of them toward one reward-preferred appearance. Under Eq. (7), the correction at a location depends on that location's state index; different initial noises give different states, so the samples of one prompt receive different corrections and keep their different compositions. DiverseGRPO, whose objective rewards diversity directly, achieves the highest LPIPS on FLUX (55.08, compared with 54.14 for the base).

E.3 Guidance-one comparisons

Table 10: Guidance-one GenEval results; LoRA and PSTune are trained separately for this setting (notation as in Table 1).

Method	GenEval			
	HPS	IR	Acc.	LPIPS
SD3.5-Medium (2.5B), guidance = 1				
LoRA, direct (Hu et al., 2022)	**26.42**	-0.354	22.1	59.46
DiffusionNFT (Zheng et al., 2026)	26.12	-0.240	**23.7**	63.22
PSTune, direct	25.27	**-0.131**	23.5	**64.53**
SDXL-base 1.0 (2.6B), guidance = 1				
LoRA, direct (Hu et al., 2022)	**27.48**	-0.056	31.6	57.65
DiffusionNFT (Zheng et al., 2026)	27.26	0.042	**33.9**	61.63
PSTune, direct	26.38	**0.151**	33.2	**62.43**

DiffusionNFT's SD3.5 and SDXL settings use guidance one, so we compare it with separately trained guidance-one LoRA and PSTune models (Table 10). PSTune has the highest ImageReward in both blocks. Relative to LoRA, its GenEval accuracy rises by 1.4 points on SD3.5 and 1.6 on SDXL, and its LPIPS rises by 5.07 and 4.78. DiffusionNFT has slightly higher accuracy, by 0.2 and 0.7 points, and PSTune leads it in ImageReward and LPIPS. PSTune also has the lowest HPS in each block while leading in ImageReward, which again separates the training reward from held-out quality. Guidance changes the scale of every score, so each comparison fixes the guidance; Tables 1 and 2 report native guidance, and Table 3 reports human judgments.

E.4 Two training objectives

Figure 17 resolves the direct and policy-gradient comparisons into the six GenEval tasks. Both objectives give positive mean accuracy and LPIPS differences over LoRA for every task and backbone. Accuracy gains are small on the near-saturated task, where the base is already close to 100%. The LPIPS gap to LoRA is larger with the direct loss on every backbone, because direct LoRA loses more variety than PG LoRA (46.63 versus 48.26 on FLUX), while PSTune keeps LPIPS near the base under both objectives.

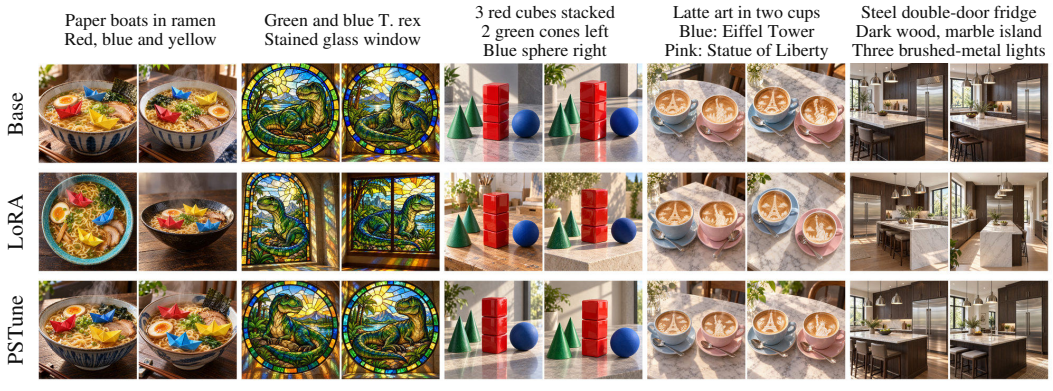


Figure 14: **Compositional prompts.** Base, LoRA, and PSTune on colored paper boats in ramen, a stained-glass dinosaur, a spatial arrangement of shapes, paired latte art, and a specified kitchen scene; each prompt has two image columns. The requests combine color or material with object identity and arrangement.

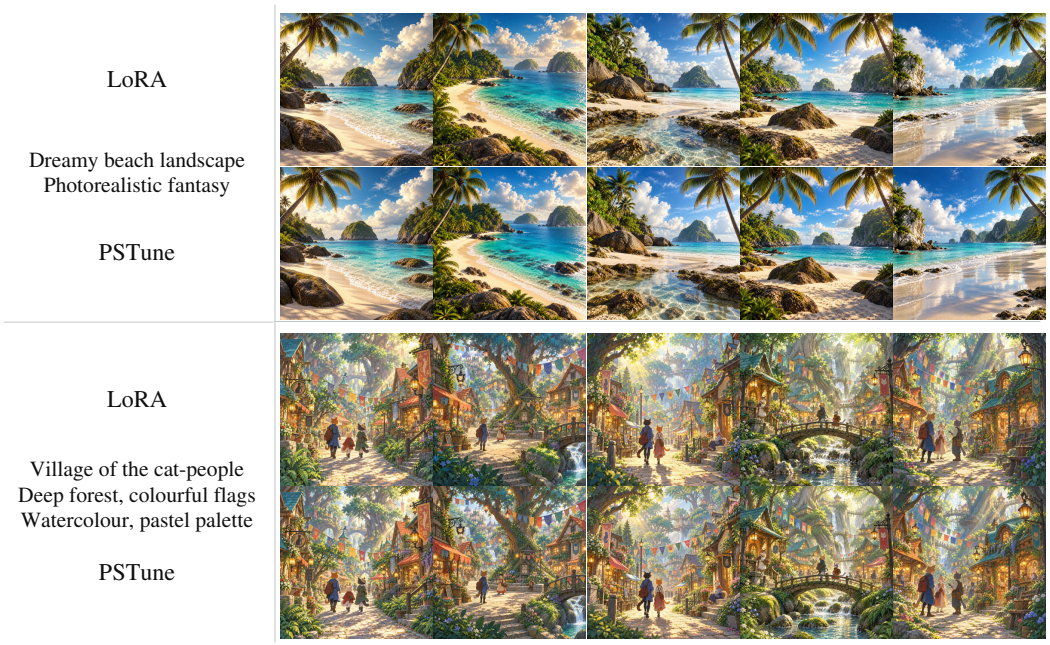


Figure 15: **Within-prompt variation.** Five LoRA and five PSTune outputs for each of two prompts (20 images). LPIPS uses all within-prompt pairs with the benchmark-specific counts of Appendix D.1.

Several of the larger accuracy gains fall on tasks where the frozen base is less accurate. A task the base already solves leaves little room for LoRA to lose, while a task the base finds hard depends on a narrow set of requested details that a shared update can overwrite; there the gap between LoRA and PSTune widens. Both objectives show positive gains on all tasks, so this effect of the update rule does not depend on how the reward gradient is estimated.

The aggregate tables show why PSTune's results barely depend on the objective while LoRA's do. Switching LoRA from the direct loss to policy gradients recovers part of its losses: GenEval accuracy rises from 63.1% to 64.5% on FLUX, from 59.8% to 61.3% on SD3.5, and from 52.0% to 53.8% on SDXL, and LPIPS rises by 1.6–2.2 points, while PG LoRA keeps a higher HPS than PG PSTune on every backbone (Tables 1 and 2). PSTune changes by at most 0.3 accuracy points and 0.2 LPIPS points between the two objectives on every backbone. How much task accuracy and variety a model keeps therefore depends mainly on where the update is stored, and only for LoRA on how the gradient is estimated.

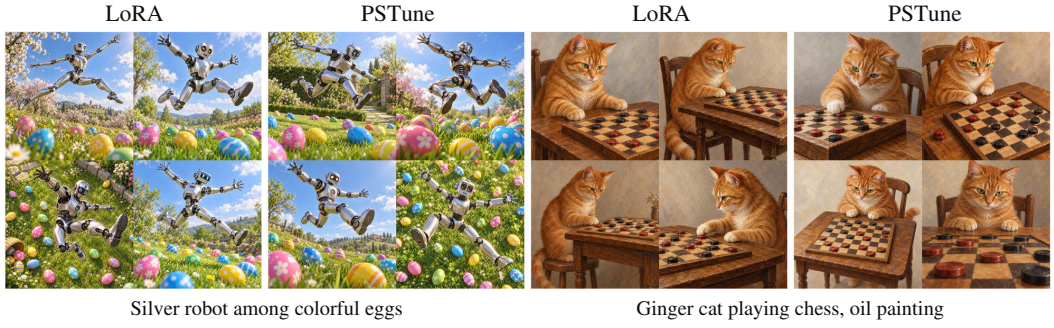


Figure 16: **Four samples per prompt and method.** LoRA and PSTune samples for a silver robot among colorful eggs and a ginger cat playing checkers in an oil painting.

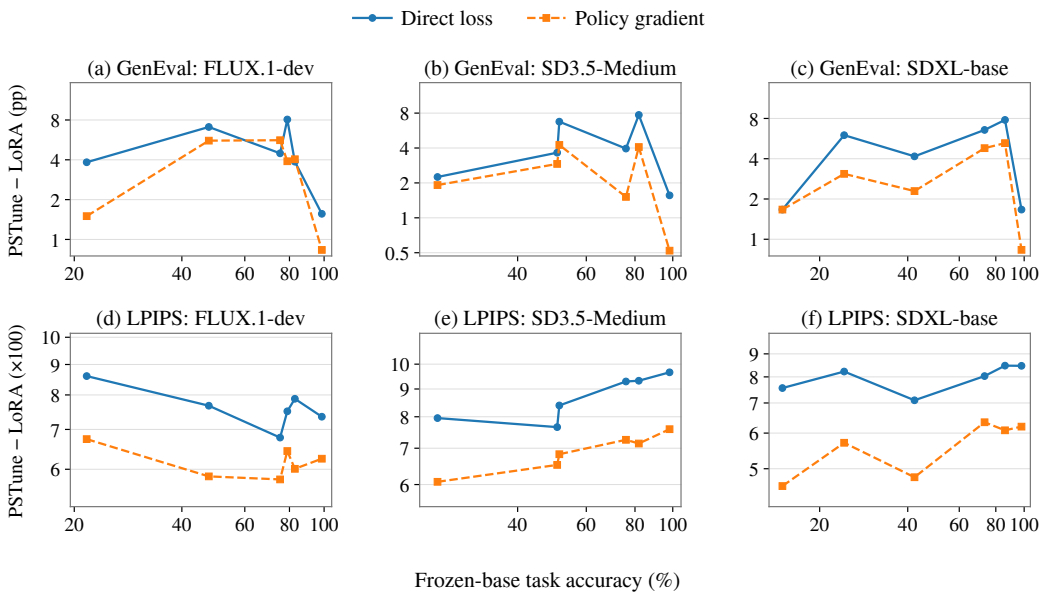


Figure 17: **Task accuracy and variety under two objectives.** PSTune-minus-LoRA differences at update 1,024, averaged within each task over three paired seeds, for FLUX, SD3.5, and SDXL. Top: GenEval accuracy in percentage points. Bottom: LPIPS $\times 100$ over all six image pairs of each of the 553 prompts. Tasks are ordered by frozen-base accuracy; both axes are logarithmic, and all plotted differences are positive.



Figure 18: **Additional object and attribute examples.** Thirty images covering paired objects, flowers, shape and color combinations, text on objects, material appearance, and indoor scenes.

The six GenEval tasks ask for single and paired objects, counts, colors, positions, and color attribution. Figure 18 shows PSTune outputs for requests of these kinds together with flowers, text on objects, materials, and indoor scenes, extending the benchmark examples of Figure 1.

F Additional Mechanism Results

F.1 Index and optimizer controls by task

Figure 19 resolves the equal-shape controls of Table 4 into GenEval tasks. A wrong prompt index and a wrong state index change the task profile in different ways, and RMSProp gives a third profile at the same update norm, so the three controls remove different parts of the transfer rule.

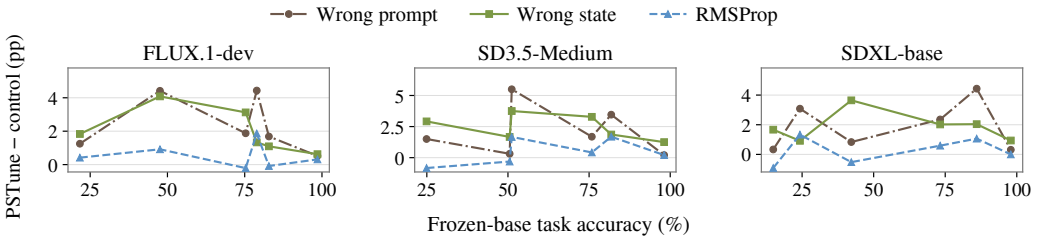


Figure 19: **Task effects of the update indices and coordinate scaling.** Direct training for 1,024 updates on each backbone. Each point is the mean of three seed-paired PSTune-minus-control accuracy differences for one GenEval task, in percentage points, with tasks ordered by frozen-base accuracy. All controls share tensor shape, initialization, forward readout, and total update norm; wrong-state controls exchange complete image-state sequences.

Table 4 gives the aggregate numbers behind these profiles. The two wrong indices cost different amounts on different backbones: a wrong prompt index lowers ImageReward more than a wrong state index on FLUX (0.204 against 0.149), and less on SD3.5 (0.143 against 0.226) and SDXL (0.175 against 0.254). Both factors of Eq. (7) therefore carry information, and their relative weight depends on how well the pooled prompt vector and the latent–velocity state separate the requests of each backbone. The text-match column separates the controls further. The two configurations with raw update features, F/F and CF/F, trail PSTune by 9.7 and 10.2 net-win points, more than either wrong index (8.4 and 5.8), and RMSProp by only 2.3. The image-wide update term thus harms text match more than a misplaced index, which fits its reach in Eq. (3): it moves every prompt-matched image regardless of which content the image requests.

F.2 Forward and update centering

Let $a_{r,b}^{f,w}$ be the GenEval accuracy of seed r and task b for forward features f and update features w, with F raw and CF centered; each of the four combinations is trained for 1,024 direct updates with the same tensor shape, data, and selection. In Figure 20(a), the update contrast is $100\,\mathrm{mean}_r(a_{r,b}^{f,CF} - a_{r,b}^{f,F})$ at each forward choice; in (b), the forward contrast is $100\,\mathrm{mean}_r(a_{r,b}^{F,w} - a_{r,b}^{CF,w})$ at each update choice; and (c) pairs $(a^{F,F} - a^{CF,F},\, a^{F,CF} - a^{F,F})$ for the raw reference and $(a^{F,CF} - a^{CF,CF},\, a^{CF,CF} - a^{CF,F})$ for the centered reference, with the same seed averaging and factor 100.

Centering the update raises accuracy on all six tasks under both forward choices (a). The forward choice interacts with the update: with raw updates, raw forward features lower accuracy on every task, while with centered updates they raise it slightly on five of six tasks (b). The image-wide update term is therefore harmful in every configuration, whereas image-wide velocity corrections help text adherence only once the image-wide update term is removed.

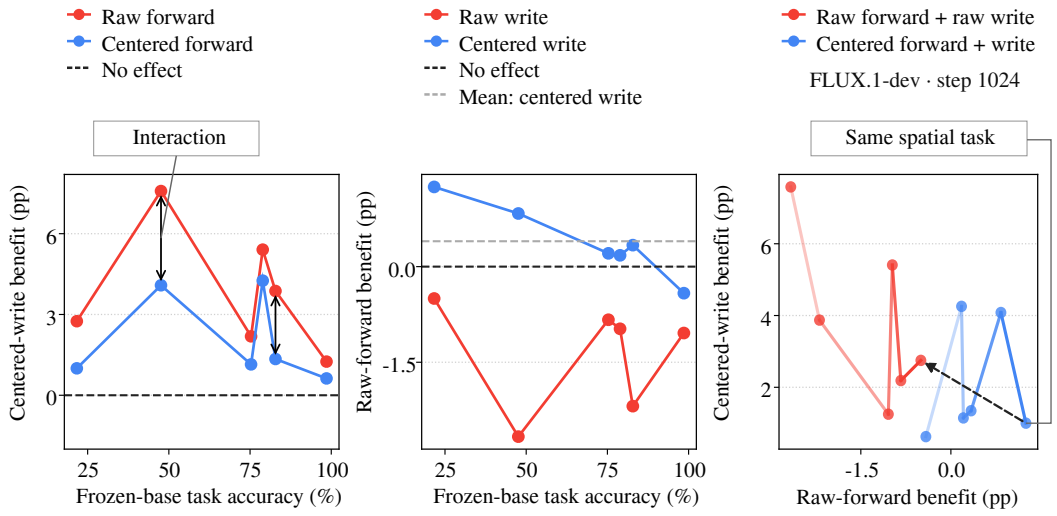
Table 11: Forward-design trade-offs after full training: PSTune (CF/CF) minus F/CF on PartiPrompts, in mean net-win points. Negative match favors raw forward features. Bold marks the largest mean in each column.

Backbone	PartiPrompts net wins (%)			
	Match	Realism	Overall	Div.
FLUX.1-dev	**-4.2**	4.5	9.7	**0.7**
SD3.5-Medium	-6.9	3.6	**11.7**	-2.3
SDXL-base	-6.1	**5.2**	9.2	-1.9

Table 11 reports human judgments of the same forward choice on all three backbones. Raw forward features lead on image–text match, by 4.2 to 6.9 points with the largest gap on SD3.5, while centered forward features lead on realism by 3.6 to 5.2 points and on overall preference by 9.2 to 11.7. Diversity changes sign across backbones and stays within 2.3 points. Each contrast uses the same frozen model, so these outcomes isolate the forward design after full training; the single-update removal of the image-wide term is analyzed in Appendix F.5. Figure 21 shows the four combinations on two GenEval prompts.

F.3 Single-source transfer at initialization and after training

Three training seeds supply 32 sources each, one per batch of 16. Each source contribution, with its original $1/16$ weight and the full-batch clipping factor, is applied separately to the same model, and



(a) Common-write removal (b) Forward-capacity effect (c) Joint benefit-cost relation

Figure 20: **Update centering and forward centering have different task effects.** FLUX.1-dev, 1,024 direct updates; six GenEval tasks and three paired seeds. (a) Centered-minus-raw update features at each forward choice. (b) Raw-minus-centered forward features at each update choice. (c) Both effects per task. Other factors are fixed within each contrast.



Figure 21: **The four forward/update combinations on two GenEval prompts.** Columns: centered/centered, centered/raw, raw/centered, and raw/raw forward/update features; rows: repeated samples for a blue clock with a white cup and a red cake with a purple chair. The clock prompt varies the arrangement of the same two objects, and the cake prompt varies the foreground object and the room.

the 553 GenEval prompts are assigned to sources within each seed; with four images per prompt, each intervention has 6,636 targets. At initialization, both adapters raise the source's HPS, and PSTune has a narrower negative tail of target changes (Figure 22); LoRA uses its full-batch AdamW preconditioner.

Figure 23 links source and target changes source by source. Its sign agreement between each source change and its targets shows how often a gain on the training image comes with a gain on the assigned images. At initialization $W = \mathbf{0}$, so the PSTune update is exactly the sum of source contributions in Eq. (11), and these outcomes show the transfer rule before any training history accumulates.

After training, at update 1,024, all 96 correct-index sources have positive HPS gains, against 75.0% with wrong-prompt and 60.4% with wrong-state updates; positive ImageReward changes occur in 63.5%, 42.7%, and 52.1% of the sources (Figure 24). Source learning and target transfer thus both depend on the index. A positive HPS change confirms that the source received a reward-improving update, and the smaller share of positive ImageReward changes shows that the training reward and an independent quality score can disagree even on the source itself, which is why we measure source learning and target transfer separately.

The other backbones repeat the single-source pattern of Table 6. A wrong prompt index keeps 33% of the correct source HPS gain on SD3.5 and 31% on SDXL, and a wrong state index 27% and 26%, while every mismatched target change is negative on both target lists. The correct target change is largest on SD3.5, at 7.40×10^{-3} on GenEval and 7.77×10^{-3} on PartiPrompts. The cached-state arm stays within 0.22×10^{-3} of the correct arm on every backbone and target list, and it is slightly higher in all six cases, so fixing the target indices at their old values neither removes nor reverses the transfer.

F.4 Predicted and regenerated target changes

For target image j assigned to source i, let $y_{ij} = Q(I_{ij}^{+}) - Q(I_j)$ be the ImageReward change after full regeneration and $d_{ij} = D_i Q_j$ the first-order prediction of Eq. (13). Each source's prediction error is $[|T_i|^{-1} \sum_{j \in T_i} (y_{ij} - d_{ij})^2]^{1/2}$ and its sign agreement is $100|T_i|^{-1} \sum_{j \in T_i} \mathbf{1}\{y_{ij} d_{ij} > 0\}$. Both use all assigned targets, and averaging per source gives every source case one weight. The wrong-prompt prediction replaces d_{ij} by the differential computed with a wrong prompt index and predicts the same correct-update outcome y_{ij}; it shares assignments, initial noise, and endpoints with the correct prediction, so the two sign agreements compare two predictions of one regenerated change.

Figure 25(a,b) compares regenerated target changes under correct and mismatched updates at update 1,024; averaged over the 96 sources, the correct update exceeds the mismatched one by 0.00698 ImageReward for prompts and 0.00764 for states. The mean source RMSE of the prediction grows from 0.00020 at initialization to 0.00208 at update 1,024, and at update 1,024 the mean sign agreement is 95.4% for the correct prediction and 48.8% for the wrong-prompt prediction. The prediction thus keeps the direction of the change while its magnitude error grows with training, consistent

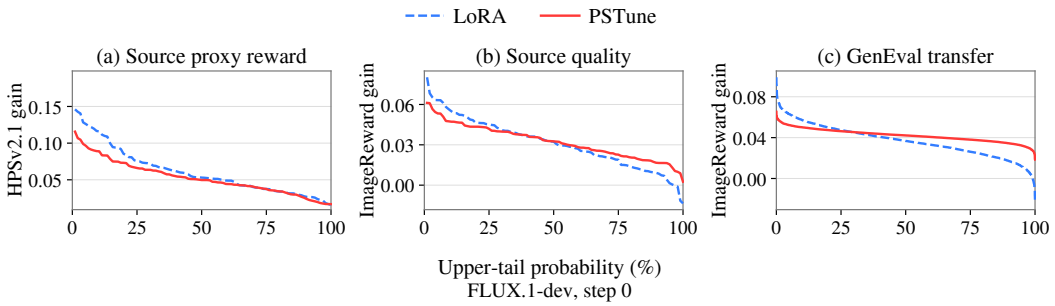


Figure 22: **Source learning and transfer at initialization.** FLUX, update 0. Each source curve contains 96 source contributions from three seeds; the target curve contains all 6,636 GenEval images. Panels show new-minus-old source HPS, source ImageReward, and target ImageReward, each sorted in descending order at upper-tail probabilities $100k/n$; HPS uses its $\times 100$ scale.

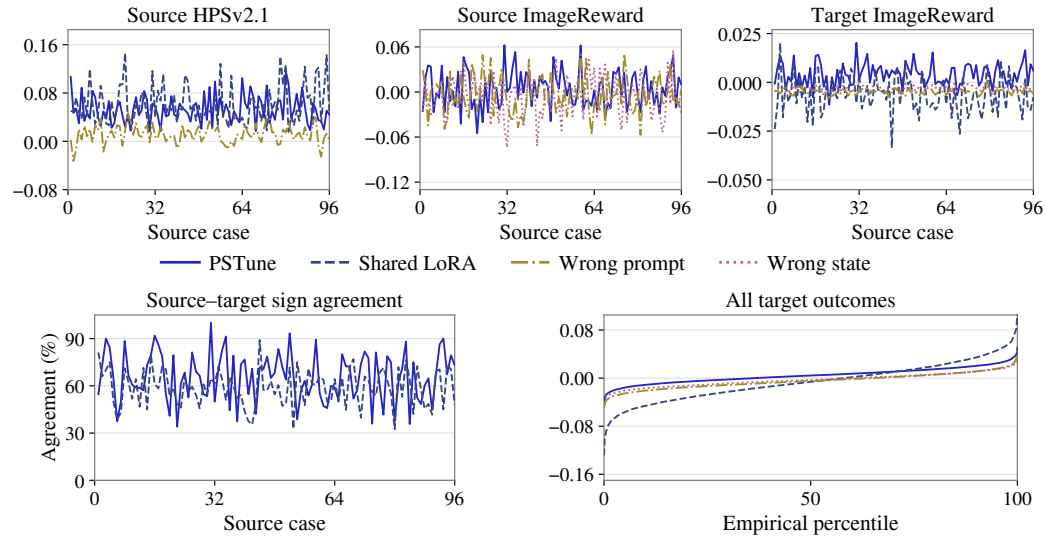
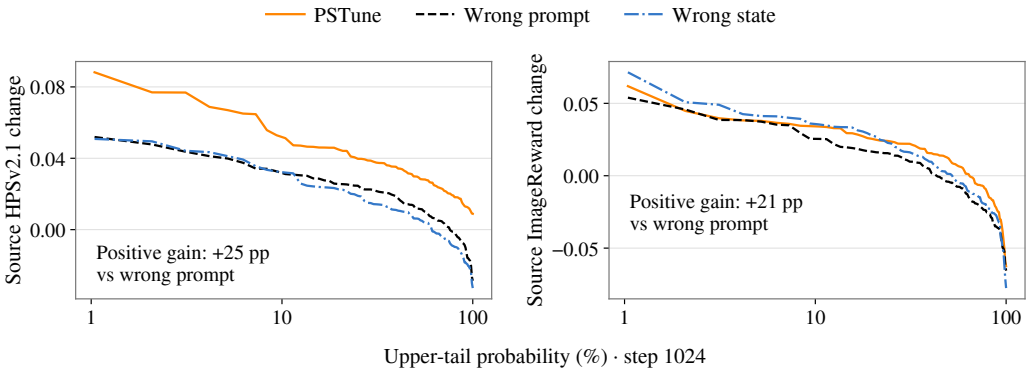


Figure 23: **Source-to-target outcomes at initialization.** FLUX, 96 sources and 6,636 GenEval images per arm. Top: source HPS, source ImageReward, and mean assigned-target ImageReward changes. Bottom: source–target sign agreement $100\,\mathrm{mean}_{j\in T_i}\,\mathbf{1}\{\Delta Q_i \Delta Q_j > 0\}$, where T_i holds every image assigned to source i and a zero product counts as disagreement, and all target changes sorted per arm. The first four panels keep source order.

with Eq. (15): the fixed-state term sets the direction, and recomputed indices and frozen-velocity responses add a growing deviation. The wrong-prompt prediction falls to 48.8%, near the 50% of an uninformative sign, because it replaces $\boldsymbol{p}_i$ by the prompt of an unrelated batch sample: its factor $\boldsymbol{p}_j^\top \boldsymbol{p}_{\text{wrong}}$ and the direction $\boldsymbol{S}_j \boldsymbol{S}_i^\top \boldsymbol{G}_i$ no longer describe the update that the target actually received. The 46.6-point gap between the two predictions therefore measures how much of the regenerated change the correct prompt factor explains.

Cached and recomputed indices. For each checkpoint, the recomputed-minus-cached difference is taken on the same prompt, noise, and source assignment before all 6,636 images are sorted (Figure 26). At update 0 the 10th and 90th percentiles are -0.000288 and 0.000264 ImageReward; at update 1,024 they are -0.003737 and 0.003460, 13.0 and 13.1 times larger. The two signs show that recomputation raises some images' scores and lowers others', and their near-symmetry explains



(a) FLUX.1-dev: source proxy (b) FLUX.1-dev: source quality

Figure 24: **Source learning under correct and mismatched indices.** FLUX at update 1,024. Each curve contains the same 96 source cases, with each arm's new-minus-old gains sorted in descending order at upper-tail probability $100k/96$ on a logarithmic axis. Correct indexing exceeds wrong-prompt indexing in the share of positive gains by 25.0 percentage points for HPS and 20.8 for ImageReward.

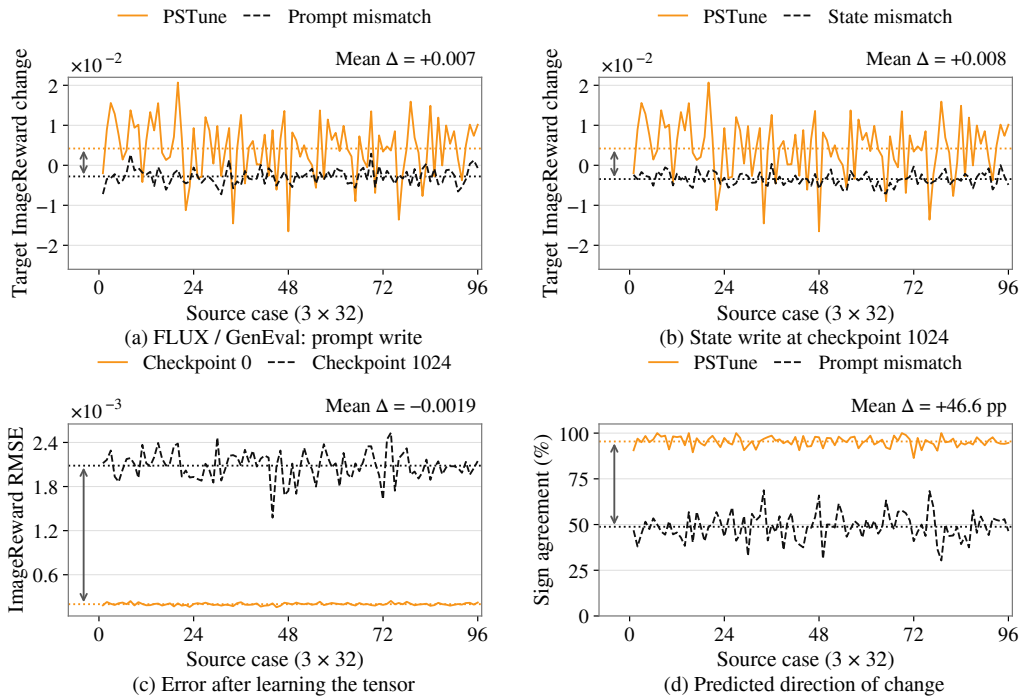


Figure 25: **The update index and the fixed-state prediction after training.** FLUX/GenEval, 96 source cases from three seeds with 6,636 assigned target images. (a,b) Mean regenerated target ImageReward change per source at update 1,024 under correct and mismatched indices. (c) Per-source RMSE between the first-order prediction and the regenerated change at updates 0 and 1,024. (d) Sign agreement of the prediction made with the correct or a wrong prompt index. Dotted lines average source cases; arrows mark mean differences.

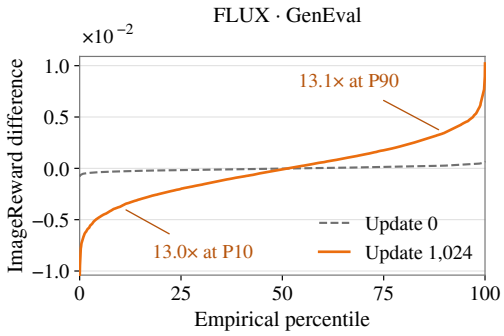


Figure 26: **Recomputed indices widen both tails.** FLUX/GenEval, 6,636 paired images per checkpoint: recomputed-minus-cached target ImageReward, sorted. The 10th- and 90th-percentile magnitudes at update 1,024 are 13.0 and 13.1 times those at update 0.

why the mean stays close to the cached arm in Table 6. The widening follows from Eq. (15): the recomputation term is bounded by $\|\boldsymbol{S}_j^+ - \boldsymbol{S}_j\|_F \|\boldsymbol{A}_j^+\|_2$, and both factors grow with training, since the trained tensor is larger and a larger update moves the trajectory, and with it the indices, further from the old one. At initialization $\boldsymbol{W} = \boldsymbol{0}$, so the term is small, which matches the narrow curve at update 0.

F.5 Removing the image-wide term in the final batch

Within each forward branch, the raw and centered updates share checkpoint, rollout, gradient, step size, and clipping factor, and paired regeneration preserves source and target noise (Appendix C.4). Figure 7 relates the update to source learning and to transfer.

Source learning. All 48 FLUX sources keep a positive HPS gain under both forward branches (panel a), so removing the image-wide term does not remove the reward signal of the source; the zero-mean term, which keeps its scale, carries it. Source ImageReward changes are mostly positive but include negative cases in both branches (panel b), again separating the training reward from an independent score on the source image itself.

Transfer. On the targets, the centered-minus-raw update improves both GenEval and PartiPrompts on all backbones and in both forward branches (panel c and Table 5). The two target lists agree: within each forward branch, the backbones rank the same on GenEval and PartiPrompts, so the effect is not specific to GenEval's short object-centric prompts. The gain is larger with raw forward features on FLUX (6.03 versus 4.64×10^{-3}) and SDXL (6.36 versus 3.88×10^{-3}). This order follows from the readout. The image-wide term stores $N\bar{\boldsymbol{g}}\bar{\boldsymbol{f}}^\top$; a raw forward readout $\boldsymbol{A}\boldsymbol{f}_{j,n} = \boldsymbol{A}(\bar{\boldsymbol{f}}_j + \boldsymbol{s}_{j,n})$ reads it through $\bar{\boldsymbol{f}}^\top\bar{\boldsymbol{f}}_j$ with the same value at every target location, whereas a centered readout reads it only through $\bar{\boldsymbol{f}}^\top\boldsymbol{s}_{j,n}$, which averages to zero over the target's locations. Removing the term therefore changes a raw-forward target more. SD3.5 is the exception (4.56 versus 5.59×10^{-3}), which shows that the size of the effect also depends on the backbone's gradients and states.

Summary of the mechanism evidence. The results of this appendix and Section 5 test the three parts of the rule separately. The product of prompt and state similarities is tested by the wrong-index controls, which keep the update norm and change only the index (Tables 4 and 6, Figures 19 and 24). The removal of the image-wide term is tested by the paired raw and centered updates of one batch (Table 5, Figure 7) and by full training with the four combinations (Table 4, Figure 20). The exactness of the fixed-state rule is tested by the sign agreement of its prediction with regenerated images and by the cached-index arm (Figures 25 and 26). Each test changes one part of the rule and keeps the others fixed, and each favors the complete rule wherever it was run: on all three backbones for the controls, and on FLUX for the prediction tests.