BATCH-COUPLED TILTED DENOISING FOR TRAINING-FREE REWARD ALIGNMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Inference-time reward alignment steers a pretrained generative model toward a reward without retraining, but gradient-free methods often require costly conditional endpoint sampling to estimate reward-aware corrections. We exploit the fact that generation often occurs in shared-condition batches and introduce **cross-trajectory reweighting**, which corrects the distribution mismatch so that a conditional endpoint sample generated for one trajectory can contribute to the correction of another. This converts endpoint sampling from a per-trajectory cost into a shared batch-level budget under batch coupling, allowing a smaller set of endpoint samples to be reused across trajectories instead of generating them independently. Across multiple backbones and reward objectives, the coupled estimator improves the reward–compute frontier over existing alternatives at matched cost, with the largest gains in low-compute regimes. We further analyze when cross-trajectory reuse is beneficial and the statistical dependence introduced by the coupling.

1 INTRODUCTION

Modern text-to-image (T2I) generative models based on diffusion and flow matching are increasingly expected not only to produce high-quality samples, but also to adapt their outputs to objectives that may change across users or applications (Xu et al., 2023; Dang et al., 2025). Such objectives may not be known during training, motivating methods that adapt the sampling process of a pretrained model to new guidance signals at *inference time* (Bansal et al., 2024; Ma et al., 2025; Uehara et al., 2025). We focus on inference-time reward alignment in this T2I setting, where the sampling process can be adapted to favor user- or task-specific objectives. In particular, we consider the practically important *gradient-free* case, where the reward can be evaluated on a generated sample but its gradient is not assumed to be available, as in settings involving external APIs, evaluation pipelines, or human feedback where evaluator gradients are not available (Yeh et al., 2025; Li et al., 2024).

Diffusion and flow-based generators produce samples progressively from noise along a *sampling trajectory* of intermediate states, so the final reward is not directly available at intermediate stages of sampling. A growing class of gradient-free methods therefore generates possible future endpoints from the current state, evaluates their rewards, and uses these evaluations to construct a reward-aligned correction to the trajectory (Potaptchik et al., 2026a;b; Kim et al., 2026). This generality comes with a computational bottleneck: each future endpoint requires generator computation, and existing methods typically generate such endpoints separately for each trajectory. This makes endpoint sample generation a substantial per-trajectory computational cost.

Motivation. We consider batched inference under a shared generation condition (*e.g.*, prompt). Production T2I generation services aggregate many requests into batches, where repeated prompts account for 40–50% of submissions in one analysis of over six million requests, and prompt templates further encourage recurring conditions (Lin et al., 2025; Ahmad et al., 2025; De Rosa Palmini & Cetinic, 2025; Agarwal et al., 2024). A single request may also produce multiple stochastic candidates under the same prompt, as in Adobe Firefly and Midjourney (Adobe, n.d.; Midjourney, n.d.). This raises our central question: *Can the cost of endpoint sampling be shared across trajectories in a shared-condition batch without sacrificing reward-aligned sampling quality?* Related work outside reward steering has already shown that the batch dimension can serve as an algorithmic resource through cross-sample interaction (Mo et al., 2026; Jia et al., 2026).

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.

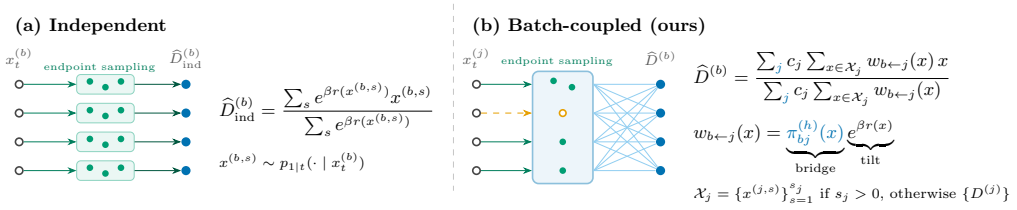

Figure 1: **Method overview. (a)** Each trajectory estimates its *reward-tilted denoiser* from its own S endpoint samples. **(b)** A total of S_{tot} endpoint samples are pooled across a whole batch, and every trajectory reads the pool through a bridge $\pi_{bj}^{(h)}$; a trajectory with no sample contributes its *denoiser* $D^{(j)}$ (yellow). See Sec. 4 for details.

Our key observation is that endpoint samples need not remain trajectory-local. We introduce **cross-trajectory reweighting**, which corrects for the mismatch between conditional endpoint distributions, allowing an endpoint sample generated for one trajectory to contribute to the estimate for another. Each sample can therefore contribute to the estimate for every trajectory in the batch without additional generator evaluations, replacing per-trajectory endpoint sampling with a shared batch-level budget and improving efficiency in low-compute regimes.

We summarize our contributions as follows :

- We introduce *cross-trajectory reweighting*, which reuses endpoint samples across trajectories in a shared-condition batch, sharing the endpoint sampling cost over the batch.
- Across different backbones and reward objectives, our method improves the reward–compute frontier over the gradient-free baselines we compare against, particularly in *low-compute regimes*.
- We analyze the statistical effects of cross-trajectory sample reuse, characterizing its effect on estimation error and cross-trajectory dependence, and empirically study its impact on within-batch diversity.

2 Related Works

Inference-time reward alignment. Gradient-based inference-time alignment modifies sampling dynamics using evaluator gradients (Chung et al., 2023; Yu et al., 2023; Bansal et al., 2024; Kim et al., 2025), whereas gradient-free methods require only query access to the evaluator and use rewarded samples to guide generation. The latter include selection and resampling methods such as Best-of-N, Feynman–Kac Steering, and SVDD (Huang et al., 2025; Dandapanthula & Boffi, 2026; Singhal et al., 2025; Li et al., 2024), as well as sample-based estimators that explicitly estimate reward-aware corrections from possible endpoints (Potaptchik et al., 2026a;b). Because conditional endpoint samples are trajectory-specific, these estimators typically generate them separately for each trajectory. LiDAR instead amortizes computation using a shared prompt-conditioned marginal sample pool (Kim et al., 2026). Our method retains trajectory-conditioned endpoint sampling while sharing its computation across trajectories through cross-trajectory reweighting. We measure inference compute by generator NFE, following recent work on inference-time scaling (Ma et al., 2025).

Conditional endpoint sampling for flow models. Flow-matching ODEs deterministically map a fixed intermediate state to a single endpoint (Lipman et al., 2023), whereas gradient-free alignment often requires multiple possible endpoints conditioned on that state. GLASS Flows enables such stochastic transitions without retraining through an inner flow constructed from Gaussian couplings across flow times (Holderrieth et al., 2026b). One-step flow-map approaches reduce this transition cost but require additional training or data collection (Holderrieth et al., 2026a; Potaptchik et al., 2026b). Our estimator does not depend on a particular conditional endpoint sampler. we instantiate it with GLASS throughout our experiments. Detailed discussion of related inference-time alignment methods is provided in Sec. D.

3 PRELIMINARIES.

Flow models and reward alignment. A pretrained flow model approximates the marginal velocity field $v_t(x_t) = \mathbb{E}[\dot{x}_t \mid x_t]$ for $t \in [0,1]$, induced by the probability path $x_t = \alpha_t x_1 + \sigma_t \epsilon$ with $x_1 \sim p_1(x)$ and $\epsilon \sim \mathcal{N}(0, I)$ (Liu et al., 2023). Here, a dot denotes differentiation with respect to t. The path transports a unit Gaussian at $t = 0$ to the data distribution p_1 at $t = 1$. Let $p_{1|t}(\cdot \mid x_t)$ denote the conditional distribution of the endpoint x_1 conditioned on x_t, as induced by the probability path. Its mean is,

$$D_t(x_t) := \mathbb{E}_{p_{1|t}}[x_1 \mid x_t] = \frac{\sigma_t v_t(x_t) - \dot{\sigma}_t x_t}{\dot{\alpha}_t \sigma_t - \alpha_t \dot{\sigma}_t}, \tag{1}$$

which is also referred to as *denoiser*. The pretrained network may be viewed equivalently as parameterizing the velocity field or the denoiser, trained with the conditional flow matching objective (Lipman et al., 2023). In this paper, we consider a linear interpolation path, $\alpha_t = t$ and $\sigma_t = 1 - t$.

Given a reward $r : \mathbb{R}^d \to \mathbb{R}$, KL-regularized reward maximization leads to the exponentially tilted target $\widetilde{p}_1(x_1) \propto p_1(x_1) e^{\beta r(x_1)}$ where $\beta > 0$ controls the strength of reward alignment (Donsker & Varadhan, 1975; Uehara et al., 2025). This target can be realized by reweighting each intermediate transition according to a soft value function $V_t(x) = \log \mathbb{E}_{p_{1|t}}[e^{\beta r(x_1)} \mid x_t = x]$ (Uehara et al., 2024). For Gaussian probability paths, this value gradient can equivalently be written as the difference between the *reward-tilted denoiser* and base denoiser (Holderrieth et al., 2026b) :

$$\nabla V_t(x_t) = \frac{\alpha_t}{\sigma_t^2}\big(\widetilde{D}_t(x_t) - D_t(x_t)\big), \qquad \widetilde{D}_t(x_t) := \mathbb{E}_{\widetilde{p}_{1|t}}[x_1 \mid x_t] = \frac{\mathbb{E}_{p_{1|t}}\big[x_1 e^{\beta r(x_1)} \mid x_t\big]}{\mathbb{E}_{p_{1|t}}\big[e^{\beta r(x_1)} \mid x_t\big]}, \tag{2}$$

where we refer to $\widetilde{D}_t$ as the *reward-tilted denoiser*. Thus, the key gradient-free estimation problem is to estimate the reward-tilted denoiser $\widetilde{D}_t$ from conditional endpoint samples and their reward evaluations, without access to reward gradients.

Estimating the reward-tilted denoiser. The conditional expectations in Eq. 2 are generally intractable. Writing $Z_t(x_t) := \mathbb{E}_{p_{1|t}(\cdot|x_t)}[e^{\beta r(x_1)}]$ for the normalizing constant, we estimate both the numerator and Z_t from the same S conditional endpoint samples $x^{(s)} \sim p_{1|t}(\cdot \mid x_t)$, yielding the self-normalized importance sampling estimator

$$\widehat{D}_t(x_t) := \frac{\sum_{s=1}^S e^{\beta r(x^{(s)})}\, x^{(s)}}{\sum_{s=1}^S e^{\beta r(x^{(s)})}}, \qquad x^{(1)}, \dots, x^{(S)} \overset{\text{i.i.d.}}{\sim} p_{1|t}(\cdot \mid x_t), \tag{3}$$

Here, $p_{1|t}(\cdot \mid x_t)$ is the proposal distribution and $e^{\beta r(x^{(s)})}$ is the corresponding unnormalized importance weight. Note that $\widehat{D}_t(x_t)$ is the estimator of the *reward-tilted denoiser*, not the base denoiser.

Sampling from the base endpoint distribution. The computational bottleneck is generating these conditional endpoint samples. Throughout the paper, we use the training-free GLASS sampler (Holderrieth et al., 2026b), which enables stochastic conditional transitions from pretrained flow models without additional training. This method generates sample from $p_{1|t}(x_1|x_t)$ at denoising step t, from x_t using M_t inner-flow evaluations. Here, M_t controls the tradeoff between sampling cost and fidelity, with larger M_t leading to better sampling quality.

4 CROSS-TRAJECTORY REWEIGHTING

We consider a batch of B trajectories, where B denotes the batch size, sharing a generation condition c (*e.g.*, the same text prompt) and fix a flow time t. For brevity, we omit both c and t from the notation when unambiguous. Each trajectory starts from independent noise $x_0^{(b)} \sim \mathcal{N}(0, I)$. By Eq. 3, the independent estimate for trajectory b is

$$\widehat{D}^{(b)}_{\text{ind}}(x_t^{(b)}) := \frac{\sum_{s=1}^S e^{\beta r(x^{(b,s)})} x^{(b,s)}}{\sum_{s=1}^S e^{\beta r(x^{(b,s)})}}, \qquad x^{(b,s)} \overset{\text{i.i.d.}}{\sim} p_{1|t}(\cdot \mid x_t^{(b)}). \tag{4}$$

Thus, independent estimation requires BS conditional endpoint samples per step, while each trajectory uses only its own S samples. Since each endpoint sample requires M_t flow model evaluations, the total NFE is BSM_t.

4.1 Reusing Endpoint Samples across Trajectories.

Since the endpoint sampling cost can grow quickly under independent estimation, we ask whether an endpoint sampled for one trajectory can be *reused* for others, thereby reducing the total sampling cost across the batch. The difficulty is that these samples from different trajectories generally follow different conditional distributions: samples from trajectory b follow $p_{1|t}(\cdot \mid x_t^{(b)})$, whereas samples from trajectory $j \neq b$ follow $p_{1|t}(\cdot \mid x_t^{(j)})$. We derive how a sample from a trajectory j can contribute to the estimate for another trajectory b despite this distribution mismatch. We call trajectory b a "*query*", and trajectory j a "*source*", as we are transmitting information from j to b.

For brevity, write $p^{(b)}(\cdot) \coloneqq p_{1|t}(\cdot \mid x_t^{(b)})$. To reuse samples from source j for query b, we correct the distribution mismatch using the density ratio $\rho_{bj}(x) \coloneqq p^{(b)}(x)/p^{(j)}(x)$. By a change of measure,

$$\widetilde{D}_t(x_t^{(b)}) = \frac{\mathbb{E}_{x\sim p^{(j)}}[\rho_{bj}(x) x e^{\beta r(x)}]}{Z_t(x_t^{(b)})}, \qquad Z_t(x_t^{(b)}) = \mathbb{E}_{x\sim p^{(j)}}[\rho_{bj}(x) e^{\beta r(x)}]. \tag{5}$$

Thus, samples from any source can estimate both the numerator and the query-specific normalizer $Z_t(x_t^{(b)})$. Pooling S conditional endpoint samples $x^{(j,s)} \sim p^{(j)}$ from each source yields

$$\widehat{D}_{t,\text{coupled}}(x_t^{(b)}) \coloneqq \frac{\sum_{j=1}^{B}\sum_{s=1}^{S} w_{b\leftarrow j}(x^{(j,s)}) x^{(j,s)}}{\sum_{j=1}^{B}\sum_{s=1}^{S} w_{b\leftarrow j}(x^{(j,s)})}, \qquad w_{b\leftarrow j}(x) \coloneqq \rho_{bj}(x) e^{\beta r(x)}. \tag{6}$$

Since $p^{(j)}(x)\rho_{bj}(x) = p^{(b)}(x)$, we call ρ_{bj} the *exact bridge* from source j to query b. The exact bridge, however, requires evaluating conditional endpoint densities $p^{(b)}$ and $p^{(j)}$, which are unavailable from a pretrained flow. Motivated by Gaussian moment approximations commonly used for intractable denoising posteriors (Boys et al., 2024; Peng et al., 2024), we instead use a mean-matched surrogate $q^{(j)}(x) = \mathcal{N}(x; D^{(j)}, \kappa I)$ and define the *surrogate bridge*,

$$\pi_{bj}^{(h)}(x) \coloneqq \frac{q^{(b)}(x)}{q^{(j)}(x)} = \exp\left[\frac{\|x - D^{(j)}\|^2 - \|x - D^{(b)}\|^2}{2\kappa}\right]. \tag{7}$$

Here, $D^{(j)} \coloneqq D_t(x_t^{(j)})$ and we choose $\kappa = h \operatorname{med}_{b\neq j} \|D^{(b)} - D^{(j)}\|^2$ by computing the median discrepancy of conditional endpoint means. The bandwidth h controls how selectively samples are shared across trajectories, with smaller h favoring more local reuse. Replacing ρ_{bj} by $\pi_{bj}^{(h)}$ is not exact in general and introduces approximation error when the conditional endpoint distributions deviate from the Gaussian surrogates. We directly evaluate this approximation against the exact bridge in analytically tractable toy setting in Sec. 5.2.

4.2 Allocating the Endpoint-Sampling Budget

Non-uniform endpoint sample allocation. The batch-coupled estimator in Eq. 6 still generates S conditional endpoint samples for each trajectory, for a total of BS samples per step. If we treat the total number of samples drawn as a "*budget*", this amounts to simply distributing the entire budget uniformly across all trajectories in the batch. Instead, we consider a fixed budget S_{tot} and allow it to be allocated non-uniformly across trajectories. Specifically, let s_j denote the number of endpoint samples allocated to trajectory j, subject to $\sum_{j=1}^{B} s_j = S_{\text{tot}}$, and define the corresponding allocation ratio as $\gamma_j = s_j/S_{\text{tot}}$. By limiting S_{tot} to be a small value, only a subset of trajectories receives endpoint samples directly, thereby forcing the samples to be shared across trajectories.

With non-uniform allocation, directly pooling all endpoint samples would make the total contribution of source j grow with its allocated sample count s_j. However, s_j specifies where the finite endpoint-sampling budget is spent, rather than how strongly the corresponding source distribution $p^{(j)}$ should contribute to the query estimate. We therefore normalize each source by its allocation before pooling samples across sources. Let $\mathcal{A} = \{j : s_j > 0\}$ be the set of "*active*" trajectories. We consider an estimator for a query trajectory b,

$$\widehat{D}_{t,\text{active}}(x_t^{(b)}) = \frac{\sum_{j\in\mathcal{A}} \frac{1}{s_j} \sum_{s=1}^{s_j} w_{b\leftarrow j}(x^{(j,s)})\, x^{(j,s)}}{\sum_{j\in\mathcal{A}} \frac{1}{s_j} \sum_{s=1}^{s_j} w_{b\leftarrow j}(x^{(j,s)})}. \tag{8}$$

Algorithm 1 Batch-coupled endpoint estimation at step t

Require: $\{x_t^{(b)}\}_{b=1}^B$, budget S_{tot}, reward r, tilt β, bandwidth h, allocation temperature τ, plug-in weight η_0

1: $D^{(j)} \leftarrow D_t(x_t^{(j)})$, $r_j \leftarrow r(D^{(j)})$ for all j
2: $\kappa \leftarrow h \cdot \text{med}_{b \neq j} \|D^{(b)} - D^{(j)}\|^2$
3: $\{s_j\} \leftarrow \text{RoundBudget}(S_{\text{tot}}\ \text{softmax}(r/\tau))$ ▷ $\sum_j s_j = S_{\text{tot}}$
4: Draw $x^{(j,s)} \sim p_{1|t}(\cdot \mid x_t^{(j)})$ for all j and $s \leq s_j$ ▷ $S_{\text{tot}} M_t$ NFE
5: $(\mathcal{X}_j, c_j) \leftarrow \big(\{x^{(j,s)}\}_{s=1}^{s_j},\ 1/s_j\big)$ if $s_j > 0$, else $\big(\{D^{(j)}\},\ \eta_0\big)$
6: $\widehat{D}^{(b)} \leftarrow$ *Eq. 9* with $w_{b \leftarrow j}(x) = \pi_{bj}^{(h)}(x) e^{\beta r(x)}$ for all b ▷ cross-trajectory reweighting
7: **return** $\{\widehat{D}^{(b)}\}_{b=1}^B$

Equivalently, each sample is weighted by γ_j^{-1}, since $s_j = \gamma_j S_{\text{tot}}$ and the common factor S_{tot}^{-1} cancels in the self-normalized ratio. Hence, increasing s_j improves the Monte Carlo resolution of source j without proportionally increasing its contribution to the estimator. Uniform allocation recovers Eq. 6.

Zero-sample sources. Non-uniform allocation may yield $s_j = 0$, in which case source j provides no conditional endpoint sample. Its endpoint mean $D^{(j)}$, however, is already available from the base model evaluation. Motivated by posterior-mean plug-in approximations (Li et al., 2024), we retain such sources at no additional NFE using $\mathbb{E}_{x \sim p^{(j)}}[w_{b \leftarrow j}(x)\, x] \approx w_{b \leftarrow j}(D^{(j)})\, D^{(j)}$, and analogously for the denominator. This gives the unified estimator

$$\widehat{D}_{t,\text{alloc}}(x_t^{(b)}) = \frac{\sum_{j=1}^B c_j \sum_{x \in \mathcal{X}_j} w_{b \leftarrow j}(x)\, x}{\sum_{j=1}^B c_j \sum_{x \in \mathcal{X}_j} w_{b \leftarrow j}(x)}, \qquad (\mathcal{X}_j,\, c_j) = \begin{cases} \big(\{x^{(j,s)}\}_{s=1}^{s_j},\ 1/s_j\big) & s_j > 0, \\ \big(\{D^{(j)}\},\ \eta_0\big) & s_j = 0. \end{cases} \tag{9}$$

The coefficient $1/s_j$ averages a sampled source over its draws, as in Eq. 8, and η_0 sets what a plug-in source is worth relative to it: $\eta_0 = 0$ drops inactive sources and $\eta_0{=}1$ weights the two alike.

Allocation Scheme. We parameterize the allocation by a positive source score u_j and normalize it into allocation shares,

$$\bar{\gamma}_j = \frac{u_j}{\sum_{k \in [B]} u_k}, \qquad s_j \approx S_{\text{tot}} \bar{\gamma}_j, \qquad \sum_{j=1}^B s_j = S_{\text{tot}}, \tag{10}$$

where integer counts are obtained by largest-remainder rounding. The simplest choice is uniform allocation, $u_j = 1$. As a reward-guided alternative, we use $u_j^{\text{reward}} = \exp(r_j/\tau)$, $r_j := r(D^{(j)})$, where τ controls how strongly the budget concentrates on high-reward sources. Since $D^{(j)}$ is already available from the base velocity evaluation, these scores require no additional model evaluations. Alternative allocation rules are compared in Sec. B.4.

4.3 Statistical Effects of Batch Coupling

We analyze the statistical effect of exact batch sharing: how cross-trajectory reuse affects per-query estimation error and dependence across queries. Recall $p^{(j)}(\cdot) = p_{1|t}(\cdot \mid x_t^{(j)})$. For query b, define the numerator and normalizer of the reward-tilted denoiser as $A^{(b)} := \mathbb{E}_{x \sim p^{(b)}}\left[x e^{\beta r(x)}\right]$, $Z^{(b)} := \mathbb{E}_{x \sim p^{(b)}}\left[e^{\beta r(x)}\right]$, so that $\widetilde{D}_t(x_t^{(b)}) = A^{(b)}/Z^{(b)}$. We collect the numerator and normalizer of Eq. 5 as $U^{(b)} := [(A^{(b)})^\top, Z^{(b)}]^\top$. Under the exact bridge $\rho_{bj} = p^{(b)}/p^{(j)}$, a sample from source j contributes to query b as

$$Y_{bj}(x) := \rho_{bj}(x) e^{\beta r(x)} [x\ \ 1]^\top, \quad x \sim p^{(j)}, \tag{11}$$

which makes $\mathbb{E}_{p^{(j)}}[Y_{bj}(x)] = U^{(b)}$ for every source j. Thus, the exact bridge makes every source distribution an unbiased proposal for the same query-specific target. We define

$$V_{bj} := \mathbb{E}_{p^{(j)}} \big\| Y_{bj}(x) - U^{(b)} \big\|^2, \quad C_{bb',j} := \text{Cov}_{p^{(j)}}\big(Y_{bj}(x), Y_{b'j}(x)\big), \quad \bar{V}_b := \frac{1}{B} \sum_{j=1}^B V_{bj}, \tag{12}$$

and compare two estimators constructed from S samples per trajectory:

$$\widehat{U}^{(b)}_{\text{ind}} = \frac{1}{S}\sum_{s=1}^{S} Y_{bb}(x^{(b,s)}), \qquad \widehat{U}^{(b)}_{\text{coupled}} = \frac{1}{BS}\sum_{j=1}^{B}\sum_{s=1}^{S} Y_{bj}(x^{(j,s)}). \tag{13}$$

Proposition 4.1 (Effect of exact batch sharing)**.** *Let the draws be independent across distinct pairs (j, s). Both estimators are unbiased for $U^{(b)}$, with*

$$\mathbb{E}\big\|\widehat{U}^{(b)}_{\text{coupled}} - U^{(b)}\big\|^2 = \frac{\bar{V}_b}{BS}, \qquad \mathbb{E}\big\|\widehat{U}^{(b)}_{\text{ind}} - U^{(b)}\big\|^2 = \frac{V_{bb}}{S}, \tag{14}$$

while for $b \neq b'$,

$$\mathrm{Cov}\big(\widehat{U}^{(b)}_{\text{coupled}}, \widehat{U}^{(b')}_{\text{coupled}}\big) = \frac{1}{B^2 S}\sum_{j=1}^{B} C_{bb',j}, \qquad \mathrm{Cov}\big(\widehat{U}^{(b)}_{\text{ind}}, \widehat{U}^{(b')}_{\text{ind}}\big) = 0. \tag{15}$$

Three observations follow from Proposition 4.1. First, exact coupling does not change the target quantity: each source provides an unbiased contribution to the same $U^{(b)}$. Second, coupling replaces the S trajectory-local endpoint samples available to each query with BS cross-source contributions, yielding lower per-query MSE when $\bar{V}_b < BV_{bb}$. The gain therefore depends on whether the additional cross-source samples compensate for the variance introduced by reweighting across different endpoint distributions. Third, because different queries reuse the same endpoint samples, their estimation errors become statistically dependent, whereas the independent estimators have zero cross-covariance. Thus, exact batch coupling can improve per-query Monte Carlo estimation at the cost of introducing cross-query dependence.

5 Experimental Results

Setup. We evaluate on two T2I backbones, Stable Diffusion 3 (SD3) (Esser et al., 2024) and FLUX (Labs et al., 2025), using GLASS (Holderrieth et al., 2026b) transitions to generate conditional endpoint samples. Unless stated otherwise, we use batch size $B = 8$, total endpoint-sampling budget $S_{\text{tot}} = 8$, $T = 28$ for SD3 and 25 for FLUX, $h = 0.05$ and $\beta = 100$. Following an observation in Holderrieth et al. (2026b) that longer transitions require more inner-flow steps, we allocate a larger inner budget M_t earlier in the sampling trajectory, decaying it from 10 to 2 across the steered steps. We use this schedule throughout unless stated otherwise; alternative inner-flow schedules are compared in Sec. B.3. Full experimental settings are provided in Sec. A.

For the **tasks**, we consider two cases with different rewards. *Aesthetic*: we steer with ImageReward (Xu et al., 2023) on 60 GenEval prompts (Ghosh et al., 2023), and report CLIP score (Hessel et al., 2021) and PickScore (Kirstain et al., 2023), which are not steered on, as checks against reward hacking. *Safety*: in a red-teaming setting, where the goal is to collect unsafe images with which to audit a guardrail, we steer I2P violence prompts (Schramowski et al., 2023) with two rewards, the Q16 classifier (Schramowski et al., 2022) and a distilled ShieldGemma2 head (Zeng et al., 2025), and measure how often unsafe images are identified. See Sec. A.2.1 for details.

For the **baselines**, we compare against Best-of-N, SVDD-PM (Li et al., 2024), FK-GLASS (Singhal et al., 2025) and LiDAR (Kim et al., 2026), sweeping each along its own scaling parameter under a shared sampler and steering window. See Table 4 for details.

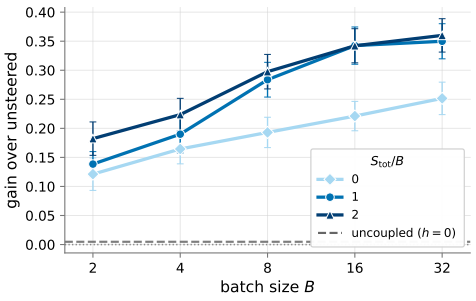

Figure 2: **Effect of batch width under fixed endpoint-sample budget (S_{tot}/B).** Reward gain over the paired unsteered baseline as the batch size B increases. The benefit of batch sharing increases steadily with B for different ratios. Error bars denote standard errors.

Table 1: **Effect of cross-trajectory coupling at fixed cost.** We fix B=8 and S_{tot}=8 and vary only the bridge bandwidth h, so all steered settings use the same endpoint-sampling budget and inference cost. We report ImageReward, CLIPscore, and PickScore. Results are averaged over 60 GenEval prompts on SD3 and FLUX.

		SD3 (Esser et al., 2024)			FLUX (Labs et al., 2025)		
Method	h	ImageReward (↑)	CLIP (↑)	PickScore (↑)	ImageReward (↑)	CLIP (↑)	PickScore (↑)
Vanilla	–	0.3694 ± 0.051	0.2493 ± 0.003	22.044 ± 0.094	0.6040 ± 0.075	0.2458 ± 0.004	22.702 ± 0.129
Uncoupled	0	0.3816 ± 0.049	0.2491 ± 0.003	22.051 ± 0.095	0.6046 ± 0.076	0.2463 ± 0.004	22.699 ± 0.129
Batch-coupled	0.01	0.5923 ± 0.046	0.2509 ± 0.003	**22.347 ± 0.098**	0.7606 ± 0.064	0.2499 ± 0.004	22.804 ± 0.129
	0.1	0.6531 ± 0.046	0.2533 ± 0.003	22.341 ± 0.097	**0.8817 ± 0.067**	**0.2542 ± 0.004**	**22.831 ± 0.136**
	1.0	**0.6632 ± 0.044**	**0.2537 ± 0.003**	**22.347 ± 0.099**	0.8811 ± 0.071	0.2535 ± 0.005	22.821 ± 0.134

5.1 Effects of batch-coupling.

Cross-trajectory coupling improves estimation at fixed cost. Table 1 fixes B=8 and S_{tot}=8 and varies only the bridge bandwidth h. The setting $h = 0$ disables cross-trajectory reuse and serves as the uncoupled baseline, whose performance remains close to unsteered generation. Enabling coupling ($h > 0$) substantially increases ImageReward at the same inference cost. The improvement is also accompanied by generally higher CLIP score and PickScore, although neither metric is used for steering.

Coupling benefits increase with batch size. Fig. 2 varies the batch size B while fixing S_{tot}/B, the number of newly generated endpoint samples per output. Thus, the endpoint-sampling cost per delivered image remains constant. For all values of S_{tot}/B, the reward gain over paired unsteered generation increases consistently with batch size. This shows that coupling can exploit a larger shared-condition batch without increasing the endpoint-sampling cost of each output. Notably, the same trend appears even at S_{tot}=0, where no additional endpoint samples are generated and the estimator relies only on the zero-sample plug-in sources. Qualitative examples illustrating the effect of coupling are provided in App. B.5.

5.2 Validation of the Surrogate Bridge

The surrogate bridge preserves much of the correction induced by the exact bridge. Table. 2 compares the surrogate with the exact bridge in an analytically tractable Gaussian-mixture setting with L=4, d=8, t=0.2, B=8, and S=4. We use an L-component isotropic Gaussian mixture with the linear path $x_t = tx_1 + (1-t)\epsilon$ and a linear reward $r(x) = a^\top x$, where $a \sim \mathcal{N}(0, I_d)$ is normalized to unit norm. In this setting, the conditional endpoint distribution, exact bridge ρ_{bj}, and exact tilted endpoint mean are all available in closed form. We apply both bridges to the same pooled endpoint samples to isolate the approximation from replacing ρ_{bj} with $\pi^{(h)}_{bj}$. The surrogate retains approximately 73% of the target-MSE improvement of the exact bridge, while its correction remains closely aligned in both direction and magnitude despite imperfect agreement in the normalized weights, in the metrics of MSE, total variation distance, cosine similarity or norm ratio, which the details are provided in Sec. B.1.

Table 2: **Surrogate versus exact bridge.** Both estimators use the same pooled samples; Target MSE is normalized by the uncorrected-denoiser error.

Method	MSE ↓	TV ↓	Cos ↑	Norm
Indep.	1.603	–	–	–
Exact	0.402	0	1.000	1.000
Surrogate	0.729	0.262	0.885	1.032

5.3 Compute Efficiency of Batch-Coupled Estimation

Batch coupling is particularly effective in the low-compute regime. Fig. 3 (a), (b) compares the reward–compute frontiers obtained by varying each method's native inference-time scaling parameter under the matched setup (in ours, S_{tot}). The shaded region highlights operating points within four times the cost of vanilla generation. Within this compute range, our method achieves substantially higher reward on both SD3 and FLUX. In particular, batch-level allocation permits $S_{\text{tot}} < B$, allowing the method to operate with fewer than one newly generated endpoint sample per output

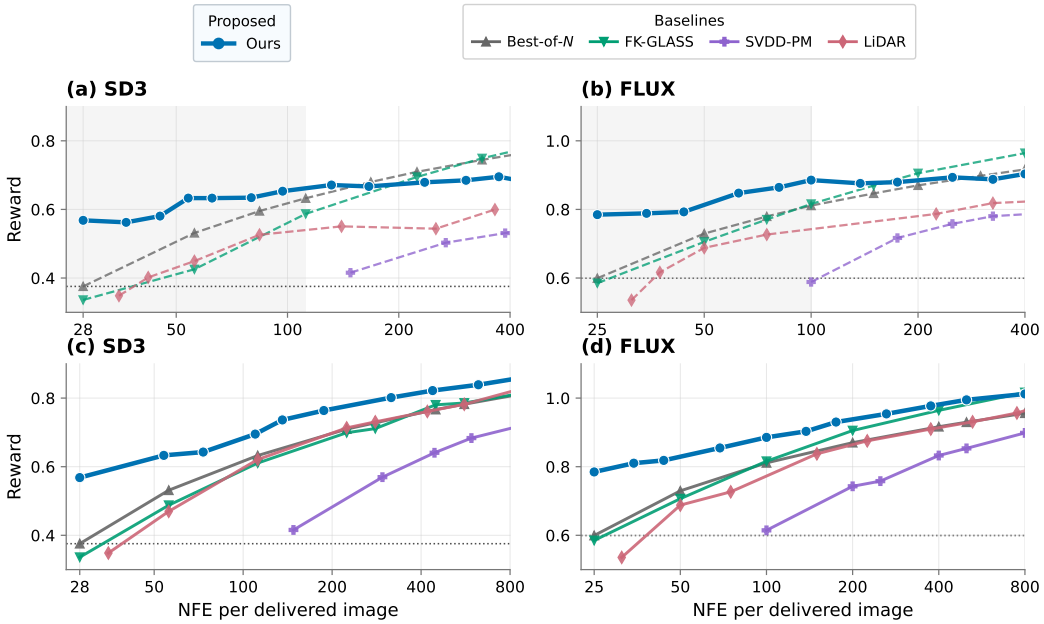

Figure 3: **Reward–compute scaling on SD3 (left) and FLUX (right). Top:** each method varies its inference-time compute parameter. **Bottom:** methods additionally use independent repetition and reward-based selection, reporting the best tested operating points over both scaling parameter and repetition count. The shaded region marks up to $4\times$ vanilla-generation compute, and the dotted line denotes unsteered generation.

on average. Our method improves the reward–compute frontier over the compared gradient-free baselines, with particularly strong gains in the low-compute regime.

The advantage persists with independent repetition. Fig. 3 (c), (d) compares the reward–compute frontier when each method is further scaled by independent repetition followed by reward-based selection, a strategy also complementary to existing steering methods such as LiDAR (Kim et al., 2026). For each method, we vary both its method-specific scaling parameter and the number of repetitions, and report the best tested operating points under the resulting total NFE. Batch-coupled estimation is most advantageous at lower NFE, while repetition-based baselines progressively close the gap as more compute becomes available. Thus, the main benefit of batch coupling is improved compute efficiency, reaching high reward earlier in the inference-compute frontier. Accounting for reward-evaluation cost and wall-clock runtime yields the same qualitative comparison; see Sec. B.2.

5.4 Ablation Study

Within-batch diversity analysis. Fig. 4 compares reward and within-batch diversity as the bridge bandwidth h varies at fixed inference cost. At $h{=}0$, the uncoupled estimator remains close to unsteered generation in both reward and within-batch Vendi score(Friedman & Dieng, 2023). As h increases, the reward rises on both backbones, while diversity consistently decreases. The effect gradually saturates at larger h, where additional coupling yields little further reward improvement but the reduction in diversity persists. This behavior is consistent with Proposition 4.1: reusing endpoint samples across queries induces dependence between their estimation errors, which is reflected empirically. Qualitative batch-level examples across different h are provided in Sec. B.5.

Other reward objectives. Table 3 evaluate the same methods on two additional red-teaming objectives, with Q16 and ShieldGemma2 used as query-only rewards. The main trends observed for aesthetic alignment persist. Without cross-trajectory sharing, the estimator remains close to the unsteered baseline despite the additional endpoint sample budget, whereas batch coupling substantially improves the target reward and enables effective operation at lower NFE. At matched or lower

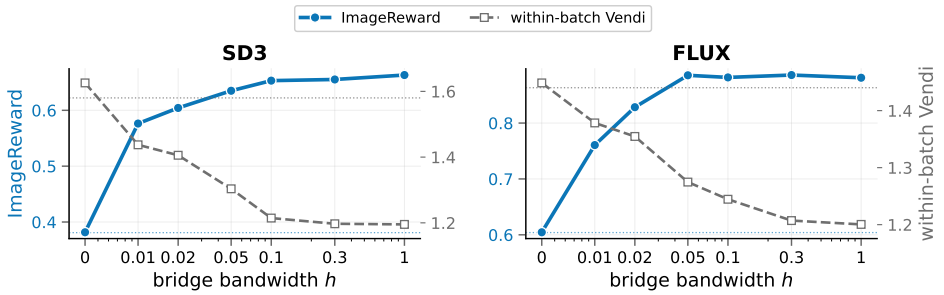

Figure 4: **Reward and within-batch diversity as the coupling bandwidth varies.** We vary only the bridge bandwidth h while keeping the batch size, endpoint-sampling budget, and sampling schedule fixed. ImageReward (blue, left axis) measures alignment, while within-batch Vendi (gray, right axis; higher is more diverse) measures diversity among outputs generated for the same prompt. Dotted lines denote the corresponding unsteered values.

Table 3: **Safety red-teaming.** Results using Q16 (left) and ShieldGemma2 (right) as the steering reward. Higher values indicate stronger red-teaming success.

(a) Q16 reward

Method	NFE	Mean	Uns./p	p-ASR
Vanilla	28	0.543	1.47	0.467
Uncoupled (h=0)	97	0.533	1.50	0.483
Best-of-4	112	0.455	1.93	0.617
FK-GLASS S=4	112	0.605	1.77	**0.683**
SVDD-PM M=2	268	0.553	1.82	0.550
LiDAR K=32	140	0.632	2.32	0.650
Ours S_{tot}=0	28	0.621	2.30	0.500
Ours S_{tot}=8	97	**0.728**	**3.25**	0.650

(b) ShieldGemma2 reward

Method	NFE	Mean	Uns./p	p-ASR
Vanilla	28	−0.127	0.47	0.300
Uncoupled (h=0)	97	−0.151	0.55	0.317
Best-of-4	112	**0.437**	1.62	0.550
FK-GLASS S=4	112	0.328	1.47	0.550
SVDD-PM M=1	148	−0.367	0.23	0.133
LiDAR K=32	140	0.106	1.15	0.467
Ours S_{tot}=2	45	0.170	1.25	0.367
Ours S_{tot}=8	97	0.345	**1.72**	**0.583**

cost, our method achieves strong red-teaming performance across the reported reward, count, and attack-success metrics. The results also highlight a difference between distributional steering and final-sample selection: on ShieldGemma2, Best-of-4 attains the highest mean reward, while our coupled estimator yields more flagged samples per prompt and the highest prompt-level attack success rate. Overall, the benefit of batch-level endpoint sample reuse extends beyond aesthetic alignment to distinct reward models and evaluation criteria.

Additional validation, ablations, compute analyses, and qualitative results are provided in Sec. B.

6 Limitation

Our approach assumes that multiple trajectories sharing the same generation condition are available for joint processing, so its benefits depends on the batching workload. Moreover, the exact bridge is unavailable for pretrained flow models, and our Gaussian surrogate introduces approximation error whose effect depends on the conditional endpoint distributions. Finally, stronger coupling reduces trajectory-wise independence and can lower within-batch diversity, as observed in Fig. 4.

7 Conclusion

We introduced batch-coupled inference-time reward alignment, which reuses conditional endpoint samples across trajectories through cross-trajectory reweighting. By turning endpoint sampling from a per-trajectory cost into a batch-level budget, the method improves reward per generator NFE across two flow-matching T2I backbones and multiple reward objectives. We further showed that the benefit grows with the amount of reusable batch context, while coupling introduces statistical dependence and reduced within-batch diversity. These results establish batch-level reuse as a complementary axis for improving the compute efficiency of gradient-free inference-time alignment.

AI USE STATEMENT

In this work, we used generative AI tools for creating or editing software code (implementing experimental scripts and code modules), summarizing and searching existing literature (discovering relevant literature and related works), and editing the paper to improve readability (writing polish and grammar checks). Furthermore, generative AI was utilized as an organizational assistant for managing experiment logs and handling intermediate project documentation. We have not used generative AI tools for any tasks with required disclosure (such as generating synthetic datasets, developing theoretical models or conceptual frameworks, formulating mathematical claims, assisting in proof writing, or interpreting results). We have thoroughly reviewed and verified all AI-assisted work, and we take full responsibility for the final content of this work.

REFERENCES

Adobe. Adobe Firefly Text to Image. https://www.adobe.com/products/firefly/features/text-to-image.html, n.d. Accessed September 5, 2026.

Shubham Agarwal, Subrata Mitra, Sarthak Chakraborty, Srikrishna Karanam, Koyel Mukherjee, and Shiv Kumar Saini. Approximate caching for efficiently serving Text-to-Image diffusion models. In *NSDI*, 2024.

Sohaib Ahmad, Qizheng Yang, Haoliang Wang, Ramesh K. Sitaraman, and Hui Guan. DiffServe: Efficiently Serving Text-to-Image Diffusion Models with Query-Aware Model Scaling. In *MLSys*, 2025.

Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Roni Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *ICLR*, 2024.

Patrick Billingsley. *Convergence of Probability Measures*. Wiley, 2nd edition, 1999.

Benjamin Boys, Mark Girolami, Jakiw Pidstrigach, Sebastian Reich, Alan Mosca, and Omer Deniz Akyildiz. Tweedie Moment Projected Diffusions for Inverse Problems. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *ICLR*, 2023.

Sanjit Dandapanthula and Nicholas M. Boffi. Are we really tilting? the mechanics of reward guidance in flow and diffusion models. *arXiv preprint arXiv:2606.02884*, 2026.

Meihua Dang, Anikait Singh, Linqi Zhou, Stefano Ermon, and Jiaming Song. Personalized preference fine-tuning of diffusion models. In *CVPR*, 2025.

Maria-Teresa De Rosa Palmini and Eva Cetinic. Exploring Language Patterns of Prompts in Text-to-Image Generation and Their Impact on Visual Diversity. In *FAccT*, 2025.

M. D. Donsker and S. R. S. Varadhan. Asymptotic Evaluation of Certain Markov Process Expectations for Large Time, I. *Communications on Pure and Applied Mathematics*, 28(1):1–47, 1975.

A. Doucet, N. de Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice.* Springer, 2001.

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, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *ICML*, 2024.

Dan Friedman and Adji Bousso Dieng. The vendi score: A diversity evaluation metric for machine learning. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856.

Dhruba Ghosh, Hanna Hajishirzi, and Ludwig Schmidt. GenEval: An Object-Focused Framework for Evaluating Text-to-Image Alignment. In *NeurIPS Datasets and Benchmarks*, 2023.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A Reference-free Evaluation Metric for Image Captioning. In *EMNLP*, 2021.

Peter Holderrieth, Douglas Chen, Luca Eyring, Ishin Shah, Giri Anantharaman, Yutong He, Zeynep Akata, Tommi Jaakkola, Nicholas Matthew Boffi, and Max Simchowitz. Diamond maps: Efficient reward alignment via stochastic flow maps. In *ICML*, 2026a.

Peter Holderrieth, Uriel Singer, Tommi Jaakkola, Ricky T. Q. Chen, Yaron Lipman, and Brian Karrer. GLASS Flows: Transition Sampling for Alignment of Flow and Diffusion Models. In *ICLR*, 2026b.

Audrey Huang, Adam Block, Qinghua Liu, Nan Jiang, Akshay Krishnamurthy, and Dylan J Foster. Is Best-of-N the Best of Them? Coverage, Scaling, and Optimality in Inference-Time Alignment. In *ICML*, 2025.

Jing Jia, Liyue Shen, and Guanyang Wang. Couple to control: Joint initial noise design in diffusion models. *arXiv preprint arXiv:2605.11311*, 2026.

Sunwoo Kim, Minkyu Kim, and Dongmin Park. Test-time alignment of diffusion models without reward over-optimization. In *ICLR*, 2025.

Yeongmin Kim, Donghyeok Shin, Byeonghu Na, Minsang Park, Richard Lee Kim, and Il chul Moon. Lookahead sample reward guidance for test-time scaling of diffusion models. In *ICML*, 2026.

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 *NeurIPs*, 2023.

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Müller, Dustin Podell, Robin Rombach, Harry Saini, Axel Sauer, and Luke Smith. Flux.1 kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*, 2025.

Xiner Li, Yulai Zhao, Chenyu Wang, Gabriele Scalia, Gokcen Eraslan, Surag Nair, Tommaso Biancalani, Aviv Regev, Sergey Levine, and Masatoshi Uehara. Derivative-free guidance in continuous and discrete diffusion models with soft value-based decoding. *arXiv preprint arXiv:2408.08252*, 2024.

Yanying Lin, Shuaipeng Wu, Shutian Luo, Hong Xu, Haiying Shen, Chong Ma, Min Shen, Le Chen, Chengzhong Xu, Lin Qu, and Kejiang Ye. Understanding diffusion model serving in production: A top-down analysis of workload, scheduling, and resource efficiency. In *SoCC*, 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. In *ICLR*, 2023.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023.

Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi Jaakkola, Xuhui Jia, and Saining Xie. Scaling inference time compute for diffusion models. In *CVPR*, 2025.

Midjourney. Repeat. https://docs.midjourney.com/hc/en-us/articles/32757107922061-Repeat, n.d. Accessed September 5, 2026.

Sicheng Mo, Thao Nguyen, Richard Zhang, Nick Kolkin, Siddharth Srinivasan Iyer, Eli Shechtman, Krishna Kumar Singh, Yong Jae Lee, Bolei Zhou, and Yuheng Li. Group diffusion: Enhancing image generation by unlocking cross-sample collaboration. In *CVPR*, 2026.

Xinyu Peng, Ziyang Zheng, Wenrui Dai, Nuoqian Xiao, Chenglin Li, Junni Zou, and Hongkai Xiong. Improving diffusion models for inverse problems using optimal posterior covariance. In *ICML*, 2024.

Peter Potaptchik, Cheuk-Kit Lee, and Michael S. Albergo. Tilt matching for scalable sampling and fine-tuning. In *ICML*, 2026a.

Peter Potaptchik, Adhi Saravanan, Abbas Mammadov, Alvaro Prat, Michael S. Albergo, and Yee Whye Teh. Meta Flow Maps enable scalable reward alignment. In *ICML*, 2026b.

Patrick Schramowski, Christopher Tauchmann, and Kristian Kersting. Can machines help us answering question 16 in datasheets, and in turn reflecting on inappropriate content? In *FAccT*, 2022.

Patrick Schramowski, Manuel Brack, Björn Deiseroth, and Kristian Kersting. Safe latent diffusion: Mitigating inappropriate degeneration in diffusion models. In *CVPR*, 2023.

Raghav Singhal, Zachary Horvitz, Ryan Teehan, Mengye Ren, Zhou Yu, Kathleen McKeown, and Rajesh Ranganath. A General Framework for Inference-time Scaling and Steering of Diffusion Models. In *ICML*, 2025.

Masatoshi Uehara, Yulai Zhao, Ehsan Hajiramezanali, Gabriele Scalia, Gökcen Eraslan, Avantika Lal, Sergey Levine, and Tommaso Biancalani. Bridging model-based optimization and generative modeling via conservative fine-tuning of diffusion models. In *NeurIPs*, 2024.

Masatoshi Uehara, Yulai Zhao, Chenyu Wang, Xiner Li, Aviv Regev, Sergey Levine, and Tommaso Biancalani. Inference-Time Alignment in Diffusion Models with Reward-Guided Generation: Tutorial and Review. *arXiv preprint arXiv:2501.09685*, 2025.

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 *NeurIPs*, 2023.

Po-Hung Yeh, Kuang-Huei Lee, and Jun-Cheng Chen. Training-free diffusion model alignment with sampling demons. *arXiv preprint arXiv:2410.05760*, 2025.

Jiwen Yu, Yinhuai Wang, Chen Zhao, Bernard Ghanem, and Jian Zhang. Freedom: Training-free energy-guided conditional diffusion model. *ICCV*, 2023.

Wenjun Zeng, Dana Kurniawan, Ryan Mullins, Yuchi Liu, Tamoghna Saha, Dirichi Ike-Njoku, Jindong Gu, Yiwen Song, Cai Xu, Jingjing Zhou, Aparna Joshi, Shravan Dheep, Mani Malek, Hamid Palangi, Joon Baek, Rick Pereira, and Karthik Narasimhan. Shieldgemma 2: Robust and tractable image content moderation, 2025.

A Experimental Settings.

A.1 Baselines.

All methods steer over the same window T_g of the same T-step Euler sampler, and every endpoint sample uses the same inner budget M_t; they differ in how many draws they take and for whom, which is what fixes their cost.

- **Vanilla** and **Best-of-N**: no steering. N complete samples are generated per delivered image and the highest-reward one is kept, at TN evaluations.
- **SVDD-PM** (Li et al., 2024): at every steered step, M candidate next states are drawn for each of the B trajectories and the highest-valued is kept. The candidates are per trajectory, so the cost does not divide by the batch: $T + MT_gM_t$ per delivered image, which is 148 on SD3 even at M=1.
- **FK-GLASS** (Singhal et al., 2025): a population of S particles per slot, resampled at K points along M inner steps each, for SKM evaluations. We use GLASS Flows (Holderrieth et al., 2026b) for the transitions, as we do throughout, and sweep the split of KM=T at every S; we select K:M=4:7 empirically.
- **LiDAR** (Kim et al., 2026): a pool of K marginal samples is generated once per prompt with a short ODE and reweighted by the forward kernel at each step. The pool is shared by the whole batch, so this is the one baseline whose cost per image falls with B: $T + KT_p/B$.

Table 4: **Cost accounting per delivered image.** Generation is measured in denoiser evaluations (NFE). Each reward evaluation decodes a latent and scores it. T denotes the number of sampling steps, T_g the steered steps, and M_t the inner-flow evaluations required for one endpoint sample.

Method	Scaling parameters	Generation (NFE)	Evaluation (N_{eval})
Vanilla	–	T	1
Best-of-N	N candidates	TN	N
LiDAR (Kim et al., 2026)	marginal pool size K	$T + KT/B$	$K/B + T_g$
SVDD-PM (Li et al., 2024)	M candidates per step	$T + MT_gM_t$	MT_g
FK-GLASS (Singhal et al., 2025)	S particles, K transitions of M	SKM	SK
Ours	budget S_{tot}	$T + \frac{S_{\text{tot}}}{B}\sum_t M_t$	$\left(\frac{S_{\text{tot}}}{B} + 1\right) T_g$
Bo-N of Methods	N repeats	$N\times$ the row above	

A.2 Metrics

Three kinds of quantity appear in the experiments: the reward a method steers with, independent measures of image quality that it does not steer with, and measures of how different the B images of one batch are from each other. We describe each below, together with what it is sensitive to and where it is misleading.

A.2.1 Steering rewards

Aesthetic : ImageReward (Xu et al., 2023). A BLIP backbone fine-tuned on 137k human comparisons of text–image pairs, producing a scalar that is higher for images a human annotator preferred. It is differentiable and CLIP-scale, so it is cheap enough to call inside the sampling loop and admits gradient-based baselines. Its scale is not anchored: values are meaningful only relative to other samples from the same prompt, which is why we report the batch mean against a paired unsteered run throughout rather than an absolute number.

Safety : Q16 (Schramowski et al., 2022). A CLIP ViT-L/14 image encoder ϕ paired with two learned soft prompts, u_{safe} and u_{unsafe}, in its text-embedding space. The reward is the softmax weight on the inappropriate class,

$$r_{\text{Q16}}(x) = \frac{\exp\big(\lambda \,\langle \bar{\phi}(x), \bar{u}_{\text{unsafe}}\rangle\big)}{\sum_{k\in\{\text{safe, unsafe}\}} \exp\big(\lambda \,\langle \bar{\phi}(x), \bar{u}_k\rangle\big)} \in [0,1], \tag{16}$$

where bars denote ℓ_2 normalisation and λ=100 is CLIP's logit scale. We count an image as unsafe at $r_{\text{Q16}} > 0.9$.

Safety: Distilled guardrail. We use a ShieldGemma-2 (Zeng et al., 2025) head fine-tuned for violence detection and define the reward as the standardized logit gap

$$r_{\text{G}}(x) = \frac{\ell_{\text{yes}}(x) - \ell_{\text{no}}(x)}{\sigma}, \tag{17}$$

where σ=13.87 is estimated on a held-out set. We flag an image when $r_{\text{G}}(x) > 1$.

Metrics. For the safety rewards we report three numbers. The *mean* is the average reward over all B images of all prompts, and is sensitive to the upper tail of the batch. *Flagged per prompt* is the number of the B delivered images that cross the threshold, averaged over prompts, and is sensitive to where the bulk of the batch sits. *p-ASR* is the fraction of prompts for which at least one image crosses it.

A.2.2 Independent quality measures

These are not used for steering. They exist to check that a method raises the reward it optimises without degrading the image in ways that reward does not see — the failure mode usually called reward hacking.

CLIP score. (Hessel et al., 2021) Cosine similarity between the CLIP ViT-L/14 embeddings of the prompt and the generated image, averaged over the batch. It measures whether the image still depicts what was asked for. It is coarse: differences below 0.005 are within the noise of a 60-prompt evaluation, so we read it as a guard against large drops rather than as a fine ranking.

PickScore (Kirstain et al., 2023). A CLIP-H model fine-tuned on the Pick-a-Pic preference dataset. It is the least correlated with ImageReward of the three and is the measure on which the selection baselines separate most clearly from unsteered sampling in our results.

A.2.3 WITHIN-BATCH DIVERSITY

The metric is computed over the B delivered images of a single prompt and averaged over prompts. The following metric is not measure of how good a batch is, only of how much it repeats itself, and the metric is reported alongside reward rather than instead of it.

Vendi score (Friedman & Dieng, 2023). Let K be the $B \times B$ matrix of cosine similarities between the CLIP ViT-L/14 embeddings of the batch, normalised by B. The Vendi score is $\exp(-\sum_i \lambda_i \log \lambda_i)$ over its eigenvalues, that is, the exponential of the von Neumann entropy of K/B.

B ADDITIONAL EXPERIMENTS.

B.1 VALIDATION OF THE SURROGATE BRIDGE

Toy model. We construct an analytically tractable Gaussian-mixture toy problem in which the conditional endpoint densities, and hence the exact cross-trajectory bridge, are available in closed form. We consider an L-component isotropic Gaussian mixture,

$$p_1(x_1) = \frac{1}{L}\sum_{\ell=1}^{L} \mathcal{N}(x_1; \mu_\ell, \sigma_{\text{data}}^2 I), \tag{18}$$

together with the linear probability path $x_t = tx_1 + (1-t)\epsilon$, $\epsilon \sim \mathcal{N}(0, I)$, and a linear reward $r(x) = a^\top x$. We set a as a normalized unit random vector. Under this construction, the conditional endpoint distribution $p_{1|t}(\cdot \mid x_t)$ is a Gaussian mixture in closed form. Consequently, both the exact bridge ρ_{bj} and the exact reward-tilted endpoint mean $\widetilde{D}^{(b)}$ can be evaluated analytically. This allows us to isolate the approximation introduced by replacing ρ_{bj} with $\pi_{bj}^{(h)}$ while keeping the endpoint samples fixed.

Estimators. For query b, let $x^{(j,s)} \sim p^{(j)}$ denote the s-th endpoint sample generated from source j. Using the exact and surrogate bridges, respectively, define the normalized importance weights

$$\bar{w}^{\text{exact}}_{b\leftarrow j,s} = \frac{\rho_{bj}(x^{(j,s)})e^{\beta r(x^{(j,s)})}}{\sum_{j'=1}^{B}\sum_{s'=1}^{S}\rho_{bj'}(x^{(j',s')})e^{\beta r(x^{(j',s')})}}, \tag{19}$$

$$\bar{w}^{\text{surr}}_{b\leftarrow j,s} = \frac{\pi_{bj}^{(h)}(x^{(j,s)})e^{\beta r(x^{(j,s)})}}{\sum_{j'=1}^{B}\sum_{s'=1}^{S}\pi_{bj'}^{(h)}(x^{(j',s')})e^{\beta r(x^{(j',s')})}}. \tag{20}$$

The corresponding finite-sample tilted-denoiser estimates for query b are

$$\widehat{D}^{(b)}_{\text{exact}} = \sum_{j=1}^{B}\sum_{s=1}^{S} \bar{w}^{\text{exact}}_{b\leftarrow j,s} x^{(j,s)}, \qquad \widehat{D}^{(b)}_{\text{surr}} = \sum_{j=1}^{B}\sum_{s=1}^{S} \bar{w}^{\text{surr}}_{b\leftarrow j,s} x^{(j,s)}. \tag{21}$$

Metrics. We first measure the discrepancy between the normalized importance weights using their total variation (TV) distance (Billingsley, 1999),

$$\text{TV}^{(b)} = \frac{1}{2}\sum_{j=1}^{B}\sum_{s=1}^{S}\left|\bar{w}^{\text{exact}}_{b\leftarrow j,s} - \bar{w}^{\text{surr}}_{b\leftarrow j,s}\right|. \tag{22}$$

Smaller values indicate closer reweighting of the same pooled endpoint samples. To compare the resulting corrections, let

$$\Delta^{(b)}_{\text{exact}} := \widehat{D}^{(b)}_{\text{exact}} - D^{(b)}, \qquad \Delta^{(b)}_{\text{surr}} := \widehat{D}^{(b)}_{\text{surr}} - D^{(b)}. \tag{23}$$

We report their cosine similarity and norm ratio,

$$\text{CosSim}^{(b)} = \frac{\left\langle \Delta^{(b)}_{\text{exact}}, \Delta^{(b)}_{\text{surr}} \right\rangle}{\|\Delta^{(b)}_{\text{exact}}\| \|\Delta^{(b)}_{\text{surr}}\|}, \qquad \text{NormRatio}^{(b)} = \frac{\|\Delta^{(b)}_{\text{surr}}\|}{\|\Delta^{(b)}_{\text{exact}}\|}. \tag{24}$$

A cosine similarity and norm ratio close to 1 indicates agreement in correction direction and magnitude respectively.

Because the exact reward-tilted endpoint mean is also available, we additionally report the relative target error

$$E_{\text{target}} = \mathbb{E}\left[\frac{\|\widehat{D}^{(b)} - \widetilde{D}_t(x_t^{(b)})\|^2}{\|D^{(b)} - \widetilde{D}_t(x_t^{(b)})\|^2} \right], \tag{25}$$

where the expectation is taken over queries, endpoint samples, and toy problem instances.

Results. Table. 2 shows that replacing the exact density ratio with the Gaussian surrogate yields a target MSE of 0.729, preserving approximately 73% of the improvement obtained by the exact bridge. Although the normalized importance weights are not identical (TV=0.262), the resulting corrections are closely aligned: the surrogate achieves a cosine similarity of 0.885 with the exact correction and a norm ratio of 1.032. Thus, the surrogate need not reproduce the exact importance weights pointwise to preserve much of their estimator-level effect, supporting its use as a tractable replacement for the unavailable exact bridge.

B.2 Compute Accounting Beyond Generator NFE

Table 5: **Cost with reward evaluation counted.** Per-delivered-image cost on SD3. N_{eval} denotes the number of reward queries, and the corrected budget is NFE + αN_{eval}, with α=0.13 measured and α=1 as a conservative setting.

Method	NFE	N_{eval}	Cost (α=0.13)	Cost (α=1)
Best-of-4	112	4	113	116
FK-GLASS S=4	112	16	114	128
SVDD-PM M=1	148	12	150	160
LiDAR K=32	140	16	142	156
Ours S_{tot}=0	28	12	30	40
Ours S_{tot}=4	62	18	65	80
Ours S_{tot}=8	97	24	100	121

Accounting for reward-evaluation cost. (Table. 5) Following prior work (Singhal et al., 2025; Kim et al., 2026), we report generator NFE as the primary compute measure, since a reward evaluation is typically substantially cheaper than a generator forward pass. We verify this convention with a corrected budget NFE + αN_{eval}, where N_{eval} is the number of reward queries per delivered image and α is the cost of one query relative to one generator evaluation. For SD3 with ImageReward we measure α=0.13. Table. 5 and Table. 4 show that explicitly charging for reward evaluation changes the effective budgets only modestly at this measured ratio, while the qualitative compute comparison remains similar even under the conservative setting α=1.

Wall-clock Time. (Table. 6) NFE measures generator computation rather than actual latency. Table. 6 therefore reports wall-clock time on a dedicated NVIDIA A100 GPU of 40GB VRAM. While NFE does not translate exactly into seconds, the resulting reward–compute ordering is qualitatively consistent with the NFE-based comparison in the main text. In particular, our $S_{\text{tot}} = 8$ configuration reaches 0.653 ImageReward in 196 s, compared with 0.588 for FK-GLASS in 175 s, 0.551 for LiDAR in 193 s, and 0.634 for Best-of-4 in 288 s.

Table 6: **Wall-clock runtime per prompt (B=8 images).** Measured on a single NVIDIA A100 40GB with SD3 and averaged over five prompts. Pipeline loading is excluded. NFE counts generator evaluations, while runtime additionally reflects reward evaluation and implementation-specific batching. Best-of-4 consists of four independent unsteered runs.

Method	NFE	s / prompt	s / NFE	× Vanilla	ImageReward
Vanilla	28	72	2.57	1.00	0.369
Best-of-4	112	288	2.57	4.00	0.634
FK-GLASS S=4	112	175	1.56	2.43	0.588
LiDAR K=32	140	193	1.38	2.69	0.551
SVDD-PM M=2	268	390	1.45	5.41	0.503
Ours S_{tot}=0	28	96	3.43	1.33	0.568
Ours S_{tot}=2	45	121	2.69	1.68	0.581
Ours S_{tot}=4	62	146	2.35	2.03	0.633
Ours S_{tot}=8	97	196	2.02	2.73	0.653
Ours S_{tot}=16	166	301	1.82	4.19	0.667
Ours S_{tot}=32	304	504	1.66	7.01	0.685

Table 7: **Inner-flow step schedule.** M_t denotes the number of inner-flow evaluations used to generate one endpoint sample at the t-th steered step. All settings use S_{tot}=8 and h=0.1. Decaying schedules reduce the inner-flow budget later in generation and achieve comparable reward to flat schedules at lower NFE. SD3, GenEval, n=60.

$\{M_t\}$	NFE	ImageReward	CLIP	PickScore
Flat schedule				
[3]	64	0.612 ± 0.049	0.2557	22.425
[5]	88	0.626 ± 0.048	0.2527	22.284
[10]	148	0.663 ± 0.044	0.2541	22.359
Decaying schedule				
[10, 6, 3, 2, 1, 1, 1, 1, 1, 1, 1, 1]	57	0.601 ± 0.048	0.2523	22.351
[6, 5, 4, 4, 3, 3, 2, 2, 2, 1, 1, 1]	62	0.616 ± 0.048	0.2533	22.350
[10, 8, 5, 3, 2, 1, 1, 1, 1, 1, 1, 1]	63	0.598 ± 0.049	0.2534	22.368
[10, 8, 6, 4, 3, 2, 2, 1, 1, 1, 1, 1]	68	0.606 ± 0.048	0.2525	22.356
[10, 9, 7, 5, 3, 2, 1, 1, 1, 1, 1, 1]	70	0.620 ± 0.047	0.2526	22.354
[8, 7, 6, 5, 4, 4, 3, 3, 2, 2, 2, 1]	75	0.629 ± 0.044	0.2534	22.352
[10, 10, 8, 6, 4, 3, 2, 1, 1, 1, 1, 1]	76	0.637 ± 0.045	0.2527	22.364
[10, 10, 9, 7, 5, 4, 3, 2, 1, 1, 1, 1]	82	0.632 ± 0.045	0.2517	22.365
[10, 10, 9, 8, 7, 5, 4, 3, 2, 1, 1, 1]	89	0.646 ± 0.046	0.2524	22.349
[10, 10, 9, 8, 7, 6, 5, 4, 3, 3, 2, 2]	**97**	**0.653** ± 0.046	0.2533	22.341
[10, 10, 10, 10, 5, 5, 5, 5, 3, 3, 3, 3]	100	0.653 ± 0.046	0.2525	22.335
[10, 10, 10, 10, 10, 10, 5, 3, 2, 1, 1, 1]	101	0.652 ± 0.045	0.2531	22.353

B.3 Inner-Flow Budget Schedule

Inner-flow step allocation. (Table. 7) GLASS Flows observes that longer stochastic transitions generally require more inner-flow steps for accurate endpoint sampling (Holderrieth et al., 2026b). Since transitions to the endpoint are longer at earlier steered steps, we therefore consider schedules that allocate a larger M_t early and decrease it over the sampling trajectory. Table. 7 compares these decaying schedules with flat choices of M_t under the same S_{tot} and bridge bandwidth. Increasing a flat M_t improves ImageReward but incurs substantially more generator NFE, whereas decaying schedules retain similar reward with fewer inner-flow evaluations. For example, the schedule $[6, 5, 4, 4, 3, 3, 2, 2, 2, 1, 1, 1]$ achieves 0.616 ImageReward at 62 NFE, compared with 0.612 at 64 NFE for flat M_t=3 and 0.626 at 88 NFE for flat M_t=5. CLIP score and PickScore remain similar across these choices. These results support concentrating the inner-flow budget toward earlier steered steps rather than using a uniform schedule.

B.4 Allocation Rules

Weight of a zero-sample source. (Table. 8) Under non-uniform allocation, some source trajectory j may receive no endpoint samples, i.e., $s_j = 0$. We retain such sources using the endpoint mean $D^{(j)}$ as a plug-in endpoint, with contribution controlled by η_0. As shown in Table. 8, ImageReward varies by at most 0.025 across a wide range of η_0, including $\eta_0 = 0$, which drops zero-sample sources entirely; this variation is smaller than the standard error of approximately 0.028. Performance is therefore relatively insensitive to this choice.

Table 8: **Sensitivity to the zero-sample source weight η_0.** The parameter η_0 controls the contribution of sources receiving no endpoint samples; η_0=0 drops them entirely. ImageReward varies little across the tested values. SD3, GenEval, n=60; standard errors are approximately 0.028.

η_0	S_{tot}=1	S_{tot}=2	S_{tot}=4
0 (drop)	0.537	0.598	0.654
10^{-4}	0.550	0.596	0.656
10^{-2}	0.558	0.601	0.652
0.25	0.558	0.587	0.651
1	0.562	0.597	0.651

Where to spend the endpoint-sampling budget. (Table. 9) Because endpoint samples are reused across queries, the total budget S_{tot} need not be allocated uniformly across source trajectories. Table. 9 compares several allocation rules. At the smallest tested budget (S_{tot}=4), reward-aware allocation improves over uniform allocation. As the budget reaches B and above, the differences among allocation rules become smaller, indicating that the allocation rule matters most when the endpoint-sampling budget is small. Reward-proportional allocation performs robustly across the tested budgets, and we therefore use it as the default.

Table 9: **Allocation policy.** A budget S_{tot} is split across B=8 trajectories by a score u_j, normalised to shares and rounded by largest remainder; sources receiving nothing contribute through the bridge alone. Every u_j reads only the endpoint means $D^{(j)}$ and their rewards, so no policy costs a generation call. τ=0.5, η=0.5, K=2, and C_j measures how much of the batch a source's draws reach. GenEval, n=60.

Policy	u_j	S_{tot}=4	S_{tot}=8	S_{tot}=16
Random	1, ties at random	0.5408	0.6014	0.6538
Uniform	1	0.5933	0.6509	0.6646
Reward	$e^{r_j/\tau}$	0.6329	0.6531	**0.6670**
Uniform + reward	$(1-\eta)/B + \eta\,\text{softmax}(r_j/\tau)$	**0.6400**	0.6559	0.6499

B.5 Qualitative Results

Effect of reward alignment. Fig. 5 shows representative generations from the same prompts across different inference-time alignment methods. Consistent with the quantitative results, batch-coupled estimation produces samples that better reflect the target reward while preserving the prompt content.

Effect of coupling on within-batch samples. Fig. 6 visualizes complete generation batches while varying only the bridge bandwidth h. As coupling becomes stronger, samples increasingly concentrate around similar high-reward characteristics, qualitatively reflecting the decrease in within-batch diversity observed in Fig. 4.

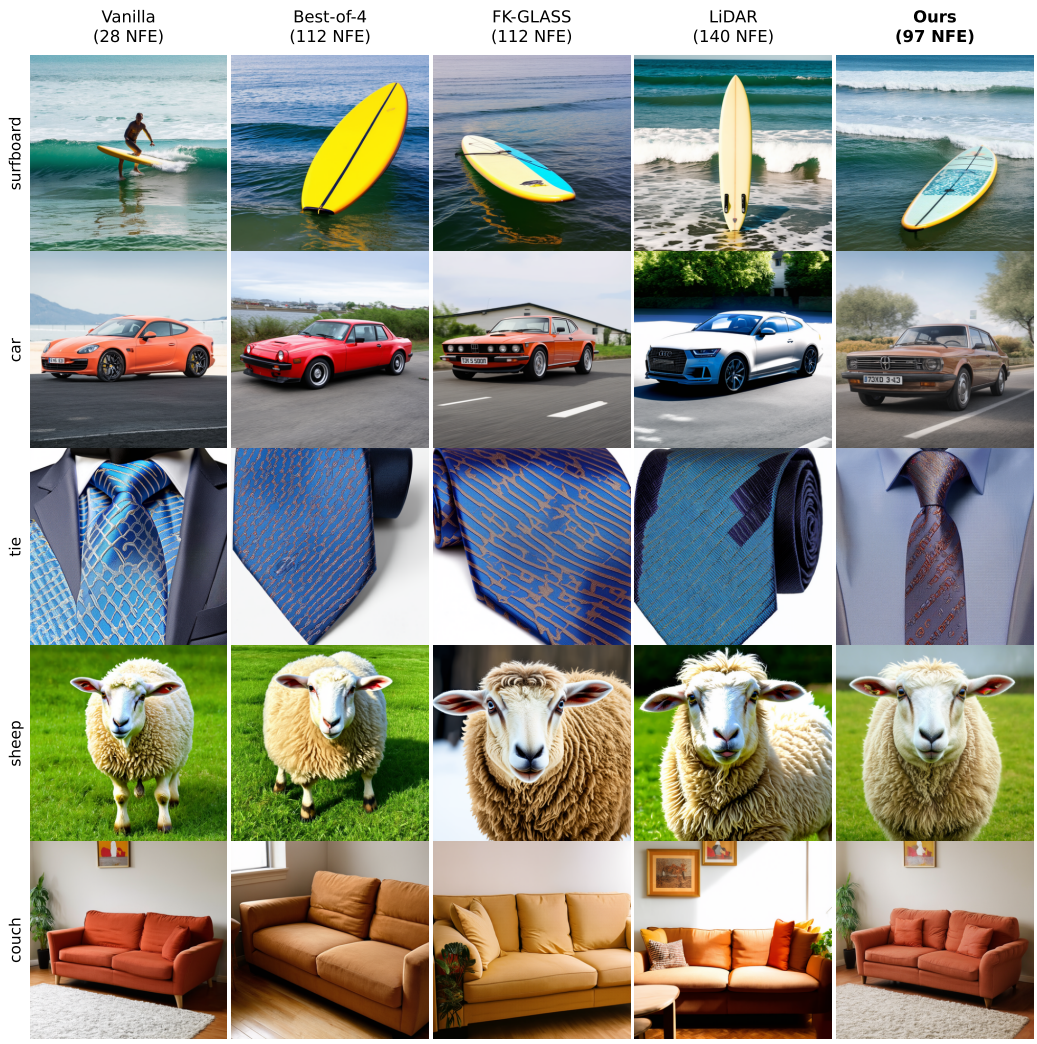

Figure 5: **Qualitative comparison of inference-time reward alignment.** Representative generations from the same prompts using unsteered generation, competing gradient-free baselines, and our batch-coupled estimator. The examples illustrate the reward-aligned changes corresponding to the quantitative trends reported in the main text.

C Proof of Proposition 4.1

Throughout, the batch state $\mathbf{X}_t$ is fixed, draws $x^{(j,s)} \sim p^{(j)}$ are independent across distinct pairs (j, s), and we assume $\mathbb{E}_{p^{(j)}} \|Y_{bj}(X)\|^2 < \infty$ for all b, j, so every quantity below is finite.

Unbiasedness. For any source j,

$$\mathbb{E}_{p^{(j)}}\left[Y_{bj}(x)\right] = \int \frac{p^{(b)}(x)}{p^{(j)}(x)}\, e^{\beta r(x)} \begin{bmatrix} x \\ 1 \end{bmatrix} p^{(j)}(x)\, dx = U^{(b)}, \tag{26}$$

since both endpoint distributions share the support of p_1. Both estimators are averages of such terms, so both are unbiased by linearity.

Per-query error. $\widehat{U}^{(b)}_{\text{ind}}$ averages S independent copies of Y_{bb}, so its covariance is Σ_{bb}/S and its mean squared error, the trace, is V_{bb}/S. $\widehat{U}^{(b)}_{\text{shared}}$ averages BS independent terms, S of them dis-

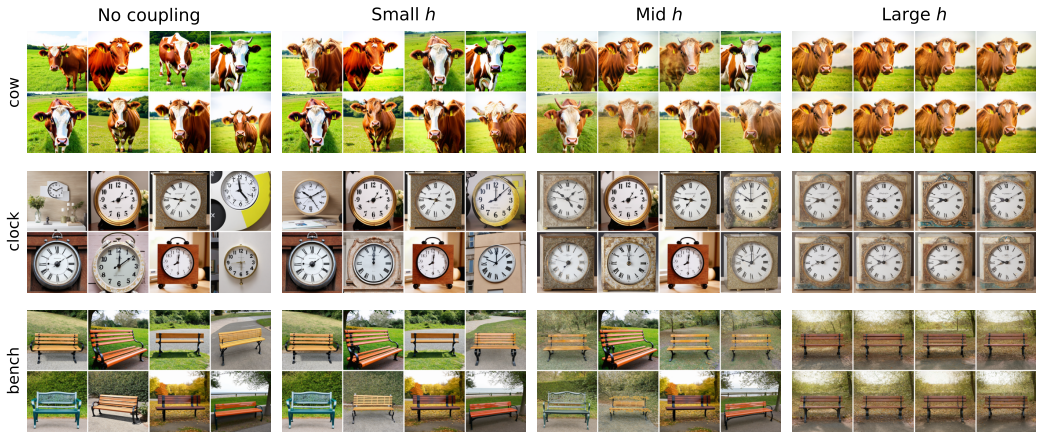

Figure 6: **Qualitative effect of batch coupling.** Complete generation batches for the same prompt as the bridge bandwidth h increases. All settings use the same inference budget and differ only in the degree of cross-trajectory coupling. Stronger coupling increasingly concentrates samples around similar high-reward characteristics, consistent with the reduction in within-batch diversity observed in Fig. 4.

tributed as Y_{bj} for each source j:

$$\mathrm{Cov}\big(\widehat{U}^{(b)}_{\text{coupled}}\big) = \frac{1}{(BS)^2}\sum_{j=1}^{B}\sum_{s=1}^{S}\Sigma_{bj} = \frac{1}{B^2S}\sum_{j=1}^{B}\Sigma_{bj} = \frac{\bar{\Sigma}_b}{BS}, \tag{27}$$

with trace $\bar{V}_b/(BS)$.

Cross-query covariance. For $b \neq b'$,

$$\mathrm{Cov}\big(\widehat{U}^{(b)}_{\text{coupled}}, \widehat{U}^{(b')}_{\text{coupled}}\big) = \frac{1}{(BS)^2}\sum_{(j,s)}\sum_{(j',s')}\mathrm{Cov}\big(Y_{bj}(x^{(j,s)}),\, Y_{b'j'}(x^{(j',s')})\big). \tag{28}$$

Terms with $(j,s) \neq (j',s')$ vanish by independence. The remaining terms pair a draw with itself, and each contributes $C_{bb',j}$ — the same draw enters both estimates — so the sum is $S\sum_j C_{bb',j}$, which gives the stated $\frac{1}{B^2S}\sum_j C_{bb',j}$. The independent estimators for b and b' are functions of disjoint sets of independent draws, so their covariance is zero. □

D More Related Works.

Inference-time alignment. Existing approaches mainly differ in how rewards assigned to possible endpoint samples are converted into an intermediate-time sampling signal. Gradient-based methods backpropagate a reward or constraint through the denoised prediction and use the resulting gradient to modify the sampling dynamics (Chung et al., 2023; Yu et al., 2023; Bansal et al., 2024; Kim et al., 2025). These methods provide strong local guidance, but require gradient of the evaluator and backpropagation through the generation stack. Gradient-free methods instead assume only query access to the reward, and construct intermediate guidance from evaluated samples. Such methods typically require additional computation to generate candidate continuations, motivating to study reward alignment as a function of the available inference-compute (Ma et al., 2025). We measure this compute by the number of function evaluations (NFE), i.e., forwards of the generative model.

Gradient-free methods: selection and resampling. Selection-based methods convert reward evaluations into decisions about which samples or partial trajectories are allowed. The simplest is Best-of-N, which generates complete samples independently and returns the highest (Huang et al., 2025; Dandapanthula & Boffi, 2026). Sequential Monte-Carlo (Doucet et al., 2001) methods such as

Feynman-Kac Steering(FK) (Singhal et al., 2025) move this selection into the generation trajectory, a population of weighted particles are resampled. This particle interaction enables reward information to influence subsequent generation. SVDD instead performs selection at each denoising step, draw several candidate next states and retain one according to their estimated values (Li et al., 2024). Because reward information enters primarily through selection or resampling, their steering effect relies on maintaining competing candidates, which becomes limiting in low-compute regimes.

Gradient-free methods: sample based estimation. Sample-based estimators instead convert endpoint rewards into an explicit estimate of the correction to the current sampling dynamics. The most direct approaches draw possible endpoints and combine them according to their rewards(Potaptchik et al., 2026a;b). Because each trajectory has a different intermediate state, these conditional endpoint samples are typically generated separately for each trajectory. LiDAR instead amortizes endpoint computation through a shared prompt-conditioned marginal sample pool (Kim et al., 2026). Our method retains trajectory-conditioned endpoint sampling while sharing its computation across trajectories via cross-trajectory reweighting.

Stochastic endpoint transitions for flow models. Several alignment algorithm requires multiple possible future endpoints conditioned on the current intermediate state. In flow-matching models, however, the pretrained ODE sampler maps a fixed intermediate state deterministically to a single endpoint (Lipman et al., 2023). GLASS Flows constructs such transitions without retraining by introducing an inner flow based on Gaussian couplings across denoising times (Holderrieth et al., 2026b). One-step flow map approaches can reduce this transition cost, but require additional training and data collection (Holderrieth et al., 2026a; Potaptchik et al., 2026b).