ACTDIFF: STRUCTURED ACTION SEARCH FOR TEST-TIME ALIGNMENT OF DIFFUSION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Test-time search aligns diffusion outputs with a reward without retraining, but the search variables determine which improvements are accessible. Noise-based branching becomes restrictive in few-step distilled and deterministic samplers. We introduce ActDiff, a framework that exposes prompt conditioning and guidance strength as structured actions at each denoising step. These actions modify the reverse transition and can be searched with gradient-free tree or particle methods. An information-theoretic analysis relates conditional reward variation to the information carried by the search variables. Across three few-step backbones, ActDiff has the highest reported target-reward means among the listed methods under IR- and HPSv3-guided search; we also evaluate it on a multi-step flow model. We use multi-reward search to balance preference criteria, reporting optimized and held-out metrics to characterize the resulting trade-offs. A separate four-backbone study finds limited within-prompt agreement between preference and alignment rewards, motivating complementary signals for candidate evaluation.

1 INTRODUCTION

Test-time search selects outputs from a frozen diffusion model (Ho et al., 2020; Song et al., 2021) using a reward model (Xu et al., 2023; Wu et al., 2023; Kirstain et al., 2023). We study which variables search should control when few-step and deterministic samplers provide limited intermediate noise branching.

Many diffusion search methods vary initial noise or intermediate stochastic innovations (Ma et al., 2025a; Li et al., 2024b; Jain et al., 2025). These choices provide useful diversity, but leave prompt conditioning and guidance fixed. Few-step samplers have a short horizon for intermediate branching, and deterministic flow samplers have no native stochastic innovations after initialization. Initial-seed search remains possible, but these settings motivate additional controls that change the denoising dynamics directly.

ActDiff treats each denoising step as a decision over structured inference-time actions. We instantiate this interface with a prompt variant and a guidance scale, both available without retraining. Monte Carlo tree search (MCTS) allocates computation among action sequences using terminal rewards. State-dependent diagnostics adapt its exploration prior using the generator's conditioning response and trajectory deviation. The same action interface also applies to particle-based propagation and can be combined with noise search.

Preference metrics emphasize different aspects of image quality, so optimizing one score can leave other criteria underrepresented. We study multi-reward search to balance preference objectives, evaluating optimized scores alongside metrics excluded from selection. A separate four-backbone study of native samples characterizes the complementary signals that motivate joint evaluation. Our contributions are: a structured action interface for diffusion inference; an information-based characterization of accessible reward variation; and evaluations of reward performance and cross-metric trade-offs across sampler regimes.

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.

2 Background and Related Work

Diffusion control and distillation. Classifier-free guidance (CFG) combines conditional and unconditional denoiser predictions to control conditioning strength (Ho & Salimans, 2021). Distillation compresses generation into a few steps (Yin et al., 2024; Zhou et al., 2025; Ge et al., 2026; Black Forest Labs, 2024); flow models support deterministic ordinary differential equation (ODE) sampling (Esser et al., 2024; Lipman et al., 2023). These developments make inference controls attractive search variables even when native intermediate noise branching is limited.

Alignment and inference-time search. Training-based alignment updates a generator for a reward or preference objective (Lee et al., 2023; Black et al., 2024; Wallace et al., 2024). Test-time methods instead search or steer candidate trajectories. Examples include Diffusion Alignment as Sampling (DAS) (Kim et al., 2025b), Feynman–Kac steering (FK-Steering) (Singhal et al., 2025), soft value-based decoding (Li et al., 2024b), beam search (Oshima et al., 2025), and Diffusion Tree Sampling (DTS) (Jain et al., 2025). Reward gradients can provide effective proposals when computationally feasible; gradient-free selection also accommodates black-box evaluators. ActDiff changes the available controls and is complementary to the choice of search procedure.

Adaptive controls and reward evaluation. Prior work adapts schedules, solvers, or guidance (Liu et al., 2024; Ni et al., 2026; Chen et al., 2024; Papalampidi et al., 2026). ActDiff jointly searches prompt and guidance actions for each test instance. Human-preference metrics and compositional alignment measures evaluate different aspects of generation (Xu et al., 2023; Ma et al., 2025b; Kirstain et al., 2023; Lin et al., 2024). Optimizing several metrics can expose trade-offs; credible assessment requires tracking which evaluators were excluded from each objective.

3 Structured Action Search

3.1 Controlled transitions and action construction

Let c_0 be the original prompt, x_T the initial latent, and $T = t_0 > \cdots > t_K = 0$ the reverse schedule. The generator parameters θ remain fixed. At state $s_k = (x_{t_k}, t_k)$, ActDiff chooses

$$a_k = (\widetilde{c}_k, \widetilde{\omega}_k) \in \mathcal{C}(c_0) \times \Omega_k(s_k), \qquad x_{t_{k+1}} = F_{\theta,k}(x_{t_k}, a_k) + \sigma_{t_k} z_k. \tag{1}$$

Here $F_{\theta,k}$ is the deterministic reverse update, $\mathcal{C}(c_0)$ contains the original prompt and proposed rewrites, and Ω_k is a finite guidance bank. Rewrites target the same entities, counts, and relations as the original prompt. The stochastic term disappears for deterministic samplers. Search may add noise choices as another action coordinate; our principal MCTS instantiation fixes the initial seed and shares the stochastic branch across structured actions at a node. All rewards evaluate the resulting image against the unchanged original prompt c_0.

Constructing the controls. The prompt bank is $\mathcal{C}(c_0) = \{c_0, \widetilde{c}^{(1)}, \ldots, \widetilde{c}^{(M-1)}\}$. A language model proposes small changes to rendering attributes such as lighting, camera phrasing, and atmosphere. The original prompt remains available as a fallback. At each decision, search selects a bank member rather than training or updating the rewriter. A prompt may therefore change between denoising stages even though the candidate bank is fixed for that instance. Semantic fidelity of the resulting bank requires validation against the original task constraints.

Guidance is a continuous control that we discretize into the finite bank $\Omega_k(s_k)$. For a denoiser exposing conditional and unconditional predictions, the usual classifier-free combination is

$$\epsilon_\theta^\omega(x_t, t, c) = \epsilon_\theta(x_t, t, \varnothing) + \omega\big[\epsilon_\theta(x_t, t, c) - \epsilon_\theta(x_t, t, \varnothing)\big]. \tag{2}$$

The sampler converts its prediction into the reverse update $F_{\theta,k}$. Prompt and guidance affect that update jointly; a tree edge specifies both coordinates. Backend-specific defaults and banks appear in Table 15.

Relation to noise branching. If $\mathcal{Z}_k$ contains candidate stochastic innovations, the general joint space is $\mathcal{B}_k = \mathcal{C}(c_0) \times \Omega_k(s_k) \times \mathcal{Z}_k$. Fixing the structured controls recovers noise-only branching; fixing the noise branch gives the structured-action search used by our MCTS instantiation. In an ODE

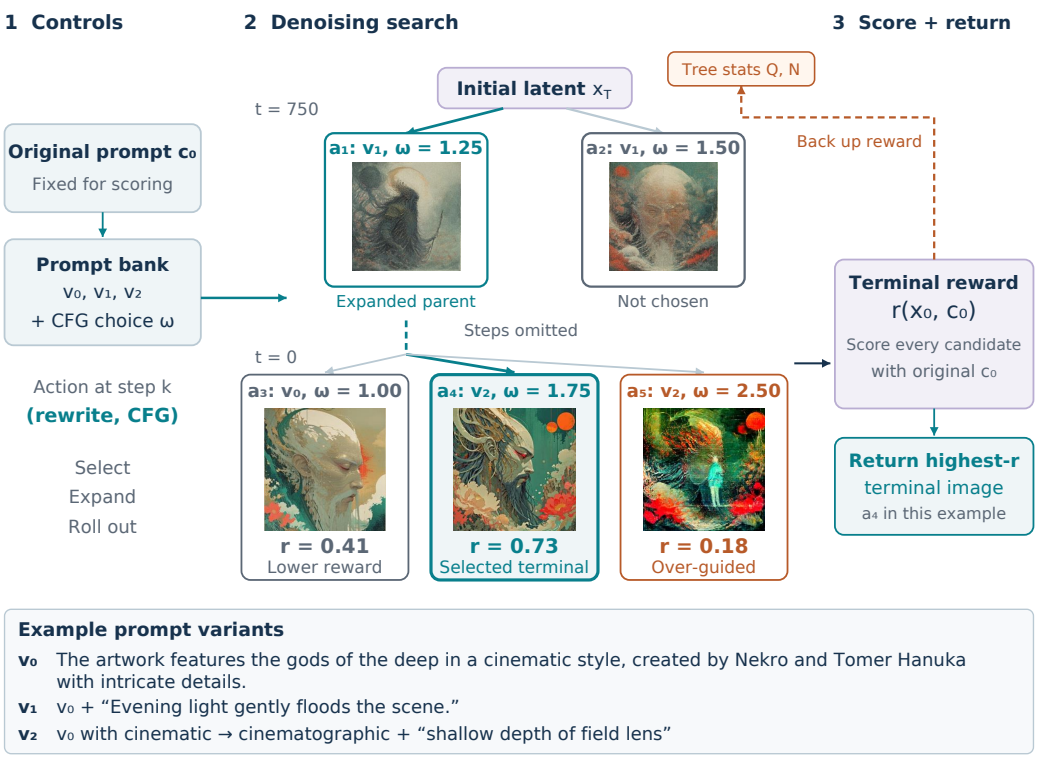


Figure 1: **Structured controls, image branches, and terminal-reward search.** Each action chooses a prompt variant and guidance value. The illustrated branches share an initial latent and use the unchanged original prompt for scoring. Terminal rewards update tree statistics, and the highest-reward terminal candidate is returned. The figure labels prompt-bank members as $v_j \in \mathcal{C}(c_0)$. Scores are illustrative; intermediate denoising steps are omitted.

sampler, only the structured coordinates remain within a trajectory. This formulation separates the diversity provided by initial seeds from the controls available after a seed has been selected.

3.2 Reward variation exposed by the action space

Fix a reference exploration policy ρ and let $\mathbf{B}$ denote a complete joint-action trajectory. For terminal reward $R = r(X_0, C_0)$, define $\mu_\rho(\mathbf{b}, c) = \mathbb{E}[R \mid \mathbf{B} = \mathbf{b}, C_0 = c]$. If $R \mid C_0 = c$ is uniformly V-sub-Gaussian, then

$$\mathbb{E}_{C_0}\left[\mathrm{Var}_{\mathbf{B}|C_0}\left(\mu_\rho(\mathbf{B}, C_0)\right)\right] \le 2V I_{\theta,\rho}(\mathbf{B}; R \mid C_0). \tag{3}$$

Bounded rewards satisfy this assumption with $V = (r_{\max} - r_{\min})^2/4$. The conditional proof is given in Appendix F. Small reward information limits the variation available to search under the reference distribution. Large information permits greater variation in the reward landscape. The factorial mutual information neural estimator (MINE) diagnostic (Belghazi et al., 2018) compares noise, prompt, guidance, and their combinations (Figure 2); it measures reward dependence under reference configurations, rather than the performance of an adaptive policy.

The reference distribution matters: changing the action bank or its sampling probabilities changes both sides of Equation 3. This is a statement about reward variation under a reference distribution, not a convergence or sample-complexity result for adaptive search. The diagnostic complements direct comparisons of generated outputs. Appendix I describes its factorial design and estimator; Appendix F supplies the assumptions and proof of the bound.

Proof sketch. For a fixed prompt c, write $\bar{r}_\rho(c) = \mathbb{E}[R \mid C_0 = c]$. The variational characterization of KL divergence, applied to the centered reward and the sub-Gaussian moment bound, gives

$$\left(\mu_\rho(\mathbf{b}, c) - \bar{r}_\rho(c)\right)^2 \le 2V D_{\mathrm{KL}}(p_{\theta,\rho}(R \mid \mathbf{b}, c) \,\|\, p_{\theta,\rho}(R \mid c)) \,. \tag{4}$$

Averaging over $\mathbf{B} \mid C_0 = c$ turns the squared mean difference into the conditional variance on the left of Equation 3, and the KL divergence into conditional mutual information. Averaging over prompts proves the result. The same reference policy defines both distributions and the expectation over actions. Because $R = r(X_0, C_0)$, data processing gives $I_{\theta,\rho}(\mathbf{B}; R \mid C_0) \le I_{\theta,\rho}(\mathbf{B}; X_0 \mid C_0)$. Image diversity alone can therefore overstate the variation relevant to a particular reward. Two action channels may change images substantially while exposing different reward landscapes; the information diagnostic evaluates the latter dependence under its specified sampling distribution.

Diagnostic design. The factorial diagnostic uses 500 HPSv2 prompts and 160,000 generated images, separately from the 200-prompt search benchmark. It crosses noise, prompt, and guidance choices on the five-value guidance grid $\{1.0, 1.25, 1.5, 1.75, 2.0\}$, with separate MINE critics for each control channel. Observed reward–control tuples form positive examples; controls shuffled within the same original-prompt group approximate the conditional product of marginals. This conditioning avoids attributing differences between prompts to the control channel. The reported implementation uses an 80/20 row-level split and three critic restarts (Appendix M); it therefore evaluates dependence within the sampled prompt pool, rather than generalization to unseen prompts. The grid and control cardinalities also delimit the interpretation of the channel comparison. MINE is an analysis tool and is not used to select actions during generation.

3.3 MCTS AND TRAJECTORY FEEDBACK

Each tree edge represents a structured action. An iteration selects a path, expands an untried action, completes a rollout, scores the terminal image, and backs up its reward. Let $Q(s, a)$ be mean terminal reward, $N(s, a)$ the edge visit count, and $N(s)$ the node visit count. Selection uses the prior-weighted PUCT rule

$$a^* = \arg\max_a \left[Q(s,a) + c_{\text{puct}} P_{\text{tf}}(a \mid s) \frac{\sqrt{N(s)}}{1 + N(s,a)} \right]. \tag{5}$$

At $s = s_k$ and $a = (\widetilde{c}, \widetilde{\omega})$, the trajectory-feedback prior P_{tf} uses two diagnostics (RMS denotes root mean square over latent dimensions):

$$u_k(s,a) = \text{RMS}(\widehat{x}_0^{\text{child}}(s,a) - \widehat{x}_0^{\text{base}}(s)), \tag{6}$$

$$d_k(s,a) = \text{RMS}(\epsilon_\theta(x_{t_k}, t_k, \widetilde{c}) - \epsilon_\theta(x_{t_k}, t_k, \varnothing)). \tag{7}$$

The prior favors informative conditioning responses while discouraging excessive deviations. Diagnostics are computed when an action is expanded and affect subsequent selection; unexpanded actions retain their relative base-prior weights. Appendix H specifies the prior and normalization, and Algorithm 1 gives the search procedure. The returned image is the highest-reward terminal sample found. The method uses forward denoising and reward evaluation without reward backpropagation.

Prior construction. Write $\bar{u}_k, \bar{d}_k$ for segment-wise reference magnitudes, and let $\mathcal{A}_{\text{exp}}(s)$ contain the actions already expanded at s. The feedback score and normalized prior are

$$\phi(s,a) = \mathbf{1}[a \in \mathcal{A}_{\text{exp}}(s)] \left[-w_u f_u\left(\frac{u_k(s,a)}{\bar{u}_k}\right) + w_d f_d\left(\frac{d_k(s,a)}{\bar{d}_k}\right) \right], \tag{8}$$

$$P_{\text{tf}}(a \mid s) = \frac{P_0(a \mid s) \exp(\phi(s,a)/\tau)}{\sum_{a'} P_0(a' \mid s) \exp(\phi(s,a')/\tau)}. \tag{9}$$

Here f_u, f_d stabilize the diagnostics, w_u, w_d weight their contributions, and τ sets the prior temperature. Reference magnitudes use the running 75th percentile of observations at the corresponding segment, with a positive floor, and are shared across candidate actions at a node. Unexpanded actions have $\phi = 0$ and retain their relative base weights. These diagnostics describe local denoising behavior; terminal reward remains the signal used to estimate action value.

Interpreting the feedback. The reference prediction $\widehat{x}_0^{\text{base}}(s)$ uses the default action at the same latent state as the candidate. Thus u_k compares local transition effects, rather than distances between unrelated initial seeds. The conditioning response d_k measures a denoiser difference; a large response need not imply better semantic alignment or a higher terminal reward. Their role is to reweight exploration through P_{tf}. Only the completed rollout reward enters Q, so the diagnostic prior and the reward-based value estimate remain distinct components of selection.

Rollout and backup. After expansion, the rollout policy completes the remaining reverse stages to produce $x_0^{(n)}$, scored as $R^{(n)} = r(x_0^{(n)}, c_0)$. For each visited edge, backup increments its count and updates its running mean:

$$N(s,a) \leftarrow N(s,a) + 1, \qquad Q(s,a) \leftarrow Q(s,a) + \frac{R^{(n)} - Q(s,a)}{N(s,a)}. \tag{10}$$

Subsequent selection uses these accumulated statistics. Search records the best terminal candidate throughout the budget and returns that image, rather than selecting an intermediate predicted clean sample. The detailed expansion and rollout procedure is given in Algorithm 1.

3.4 Integration with particle search

The same controls can be sampled during particle propagation. In a population of P particles, particle i draws an action $A_k^{(i)}$ from its proposal policy and advances through Equation 1. The underlying particle method supplies an incremental reward potential $G_{k+1}^{(i)}$, giving normalized weights

$$w_{k+1}^{(i)} = \frac{w_k^{(i)} G_{k+1}^{(i)}}{\sum_{j=1}^{P} w_k^{(j)} G_{k+1}^{(j)}}. \tag{11}$$

Resampling retains complete particle histories, including their selected controls. Thus, structured actions expand the propagation choices while weighting and resampling remain those of the underlying method. Any proposal correction is method-specific. This construction does not assert exact sampling from a reward-tilted distribution; its empirical effect is evaluated through the native-method and +ActDiff comparisons in Table 8. Expanded formulations appear in Appendix A.

4 Experiments

Setup. The main benchmark uses 200 HPSv2 prompts and four-step FLUX.1-schnell, SiD-SD3.5-Large, and SenseFlow-SD3.5-Large, alongside multi-step SD3.5-Large (Wu et al., 2023; Black Forest Labs, 2024; Zhou et al., 2025; Ge et al., 2026; Esser et al., 2024). We abbreviate the distilled SD3.5-Large backbones as SiD-SD3.5 and SenseFlow-SD3.5, and shorten backbone names to FLUX, SiD, and SenseFlow where unambiguous. We report ImageReward (IR), HPSv3, HPSv2, and PickScore, distinguishing the search objective from excluded metrics. Baselines include best-of-N (BoN) (Stiennon et al., 2020), Search over Paths (SoP), a genetic algorithm (GA), DTS, DAS, FK-Steering, and beam search. Benchmark NFEs count diffusion-backbone evaluations. Under the two-branch CFG forward-equivalent convention, CFG $= 1$ uses one conditional evaluation, whereas CFG > 1 uses conditional and unconditional evaluations. Decimal NFEs are averages of these integer counts over evaluated prompts with different selected guidance scales. Tables 2, 6, and 7 instead report configured batched-invocation costs from Table 17, counting the paired CFG branches as one batched invocation. Auxiliary costs are itemized in Appendix N, including a separate four-prompt SiD wall-clock and memory study (Appendix N.1). Main-benchmark prompt rewriting uses Qwen3-4B (Yang et al., 2025). Detailed settings and per-method configuration tables are in Appendix M.

Metrics and comparison scope. ImageReward is a BLIP-based preference model; HPSv2 and PickScore use CLIP-based representations, while HPSv3 uses a vision–language model (Xu et al., 2023; Wu et al., 2023; Kirstain et al., 2023; Ma et al., 2025b). All evaluate the original prompt. An evaluator counts as held out only when excluded from the corresponding search objective. In particular, PickScore is held out in the IR+HPSv3 benchmark but optimized in the IR+HPSv3+PickScore benchmark. The benchmark and reward-orthogonality study use different cohorts; their sample counts and baselines are not pooled. Reward tables report means to three decimal places, with rankings based on the displayed precision. We compare means at each method's reported budget; these benchmark runs do not include paired uncertainty estimates or end-to-end timings.

4.1 Benchmark Results

Single-reward performance. Under ImageReward-guided search, ActDiff achieves IR scores of 1.687, 1.603, and 1.640 on FLUX, SiD, and SenseFlow, respectively (Table 4). HPSv3-guided search

Table 1: Single-reward search: optimized scores from separate IR/HPSv3 runs on 200 HPSv2 prompts. Bold marks the best displayed score.

Method	FLUX.1-schnell		SiD-SD3.5		SenseFlow-SD3.5	
	IR	HPSv3	IR	HPSv3	IR	HPSv3
Distilled Baseline	1.163	13.558	1.180	12.340	1.269	12.220
DAS	1.488	–	1.518	12.980	1.488	14.030
FK-Steering	1.515	14.990	1.479	13.490	1.527	13.680
BoN	1.583	14.732	1.513	13.003	1.544	13.990
Beam	1.557	14.836	1.462	12.850	1.349	13.320
SoP	1.446	14.028	1.470	12.670	1.343	12.650
GA	1.537	14.218	1.379	12.730	1.350	12.910
DTS	1.251	13.490	1.245	13.644	1.281	12.023
DTS*	1.363	14.670	1.490	12.833	1.476	13.885
Dynamic CFG	1.402	13.803	1.236	12.380	1.346	12.470
ActDiff	**1.687**	**15.528**	**1.603**	**13.750**	**1.640**	**14.939**

Higher scores are better. Budgets differ; Tables 4 and 5 give NFEs, excluded metrics, and the objective-specific DAS/FK-Steering variants. DTS* is the separately reported variant; Dynamic CFG is guidance-only. A dash means unavailable. The MINE diagnostic measures reward dependence, not a guarantee of search improvement.

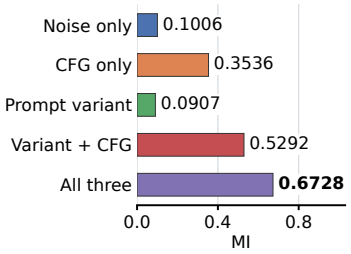


Figure 2: MINE-estimated conditional reward information under factorial reference configurations (Belghazi et al., 2018).



Figure 3: Four-method comparison at matched budgets and identical seeds, across all three backbones and both reward objectives. Appendix B.3 continues each of these comparisons on a second prompt and adds the remaining backbone–objective pairs.

also improves the target reward (Table 5). SD3.5-Large results demonstrate the use of structured actions with an ODE sampler (Table 6); noise-dependent baselines require stochastic variants or initial-seed variation.

Table 1 brings the target-reward comparisons together across all three distilled backbones. The IR and HPSv3 columns come from separate searches under the respective objective; they are not

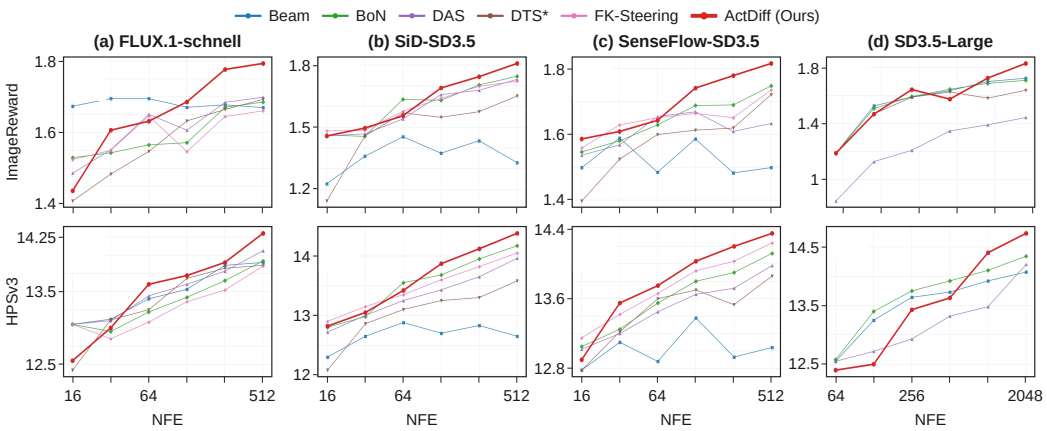


Figure 4: Reward versus backbone NFEs; top row ImageReward-guided, bottom row HPSv3-guided, each column sharing its NFE axis. ActDiff keeps improving as the budget grows, whereas Beam flattens or oscillates on SiD-SD3.5 and SenseFlow-SD3.5. Panel (d) uses an ODE sampler, where noise-dependent baselines are unavailable. Budget and seed-search discussion: Appendix B.1.1.

two measurements of a single jointly optimized run. Across these settings, ActDiff has the highest reported target-reward means among the displayed methods. The complete tables also report excluded metrics and variants of the particle methods, allowing favorable target scores to be assessed alongside cross-metric changes. Figure 3 compares all four methods on two settings; Appendix B.3 continues those comparisons and covers the remaining backbones and objectives.

When action augmentation hurts. Adding structured controls to another search method is not uniformly beneficial. In Table 4, SiD SoP's IR falls from 1.470 to 1.406. SenseFlow SoP's IR rises from 1.343 to 1.385, but HPSv3 falls from 12.315 to 12.122 and PickScore from 22.650 to 22.593. Thus, compatibility with a search algorithm does not imply an improvement on every backbone or metric.

Scaling and control ablations. Figure 4 compares reward against backbone NFEs under ImageReward- and HPSv3-guided search on four backbones. ActDiff's reward increases with budget, while its relative advantage depends on the backbone, objective, and budget. Beam, for example, scores higher at the smallest displayed budget in the FLUX ImageReward panel. Appendix B.1.1 discusses the budget-dependent behavior and seed-search configurations. Restricting the action space costs most of the gain: on SD3.5-Large, removing guidance search lowers IR from 1.649 to 1.493, and removing prompt search lowers it to 1.487 (Table 12). The complete control, bank-size, exploration, prior, and rewriter ablations are in Appendix B.4.

4.2 Application to SMC and Human Evaluation

Structured actions in particle propagation. We evaluate ActDiff in DAS, an SMC-based diffusion sampler, and FK-Steering, a Feynman–Kac particle method (Kim et al., 2025b; Singhal et al., 2025). The native method supplies its reward potentials, particle weights, and resampling rule; +ActDiff adds prompt–guidance choices during propagation (Section A.4). These comparisons test the action interface beyond MCTS. Table 8 reports native and augmented methods under four reward objectives, with 256 NFEs stated for each configuration.

Dependence on the reward objective. Under HPSv3 guidance, +ActDiff increases HPSv3 in all six method–backbone pairs. With IR+HPSv3, both optimized metrics increase for DAS and FK-Steering on all three backbones: IR gains range from 0.016 to 0.076 and HPSv3 gains from 0.090 to 0.525. With IR+HPSv3+PickScore, 17 of the 18 optimized-metric comparisons improve. The exception is SiD FK-Steering: PickScore falls from 22.818 to 22.550, while held-out HPSv2 falls from 0.309 to 0.308. The appendix separates optimized and held-out changes in Table 14. These results show that the structured action interface transfers to particle search, with gains and regressions depending on the reward objective and backbone.

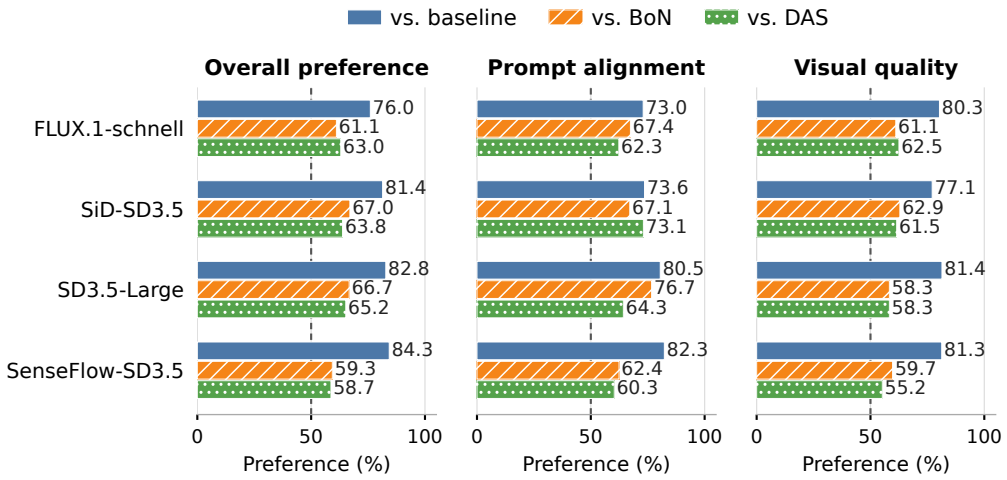


Figure 5: Human preference rates of ActDiff against baseline sampling, Best-of-N (BoN), and Diffusion Alignment as Sampling (DAS). Each setting contains 50 attempted comparisons. Missing responses are excluded, while no-preference responses are counted as half a win. The dashed line indicates the no-preference level of 50%. The protocol is in Appendix B.2.

Table 2: Multi-reward search under IR + HPSv3 and IR + HPSv3 + PickScore, with representative selection and particle baselines. HPSv2 is held out for both objectives; PickScore is held out only for IR + HPSv3. Configured NFEs follow Table 17; ActDiff reports an estimated mean. The complete comparison is in Tables 20 and 21.

	FLUX.1-schnell				SiD-SD3.5				SenseFlow-SD3.5			
Method	Cfg. NFE	IR	HPSv3	HPSv2	Pick	Cfg. NFE	IR	HPSv3	HPSv2	Pick	Cfg. NFE	IR	HPSv3	HPSv2	Pick
IR + HPSv3															
BoN	128	1.404	14.636	0.312	22.809	128	1.422	13.786	0.309	22.832	128	1.412	13.898	0.310	22.830
FK-Steering	128	1.447	14.592	0.313	22.820	128	1.400	13.470	0.311	22.816	128	1.424	13.546	0.314	22.817
ActDiff	≈ 103	1.467	15.024	0.314	22.853	≈ 103	1.455	14.109	0.310	22.845	≈ 103	1.450	14.408	0.310	22.843
IR + HPSv3 + PickScore															
BoN	128	1.565	13.414	0.313	22.845	128	1.569	12.828	0.312	22.901	128	1.562	12.865	0.311	22.981
FK-Steering	128	1.540	13.526	0.311	22.844	128	1.537	12.674	0.309	22.818	128	1.567	12.593	0.310	22.813
ActDiff	≈ 103	1.629	13.856	0.313	22.909	≈ 103	1.625	12.851	0.311	22.910	≈ 103	1.644	12.976	0.313	22.998

Pick denotes PickScore. NFE counts batched denoiser invocations per trajectory step, with CFG branches batched together. ActDiff has a configured worst-case bound of 140 NFEs. Reward-model evaluations are excluded.

Human judgments. Figure 5 summarizes a separate study of 600 image pairs: 50 GenAI-Bench prompts on each of four backbones, with three comparators per prompt. Eight evaluators assess overall preference, original-prompt alignment, and visual quality. Method identities are concealed and image order is randomized. Each image pair receives one judgment per question, with an explicit no-preference option. For wins W, losses L, and ties N_{tie}, the reported preference rate is $\frac{W+0.5N_{\text{tie}}}{W+L+N_{\text{tie}}}$. Across backbones, overall preference for ActDiff is 76.0–84.3% against baseline sampling, 59.3–67.0% against BoN, and 58.7–65.2% against DAS. With one evaluator per pair, inter-rater agreement is unavailable, and valid counts depend on missing responses. The full protocol is in Appendix B.2.

4.3 Balancing Preference Criteria with Multi-Reward Search

Joint preference objective. We search with IR+HPSv3 and IR+HPSv3+PickScore to balance preference criteria during candidate selection. HPSv2 is held out under both objectives; PickScore is held out only under IR+HPSv3. The separate orthogonality study examines agreement between preference and prompt-alignment scores.

Table 3: Reward complementarity across four backbones. Reported within-prompt Pearson correlations and nominal 95% confidence intervals from 3,200 native samples over 200 prompts.

Reward pair	Pearson r	95% CI
HPSv3–ImageReward	0.32	$[0.25, 0.38]$
ImageReward–VQAScore	0.28	$[0.18, 0.37]$
HPSv3–VQAScore	0.11	$[0.05, 0.17]$

Multi-reward objectives. Table 2 summarizes the multi-reward results. The objectives are IR+HPSv3 and IR+HPSv3+PickScore. In each setting, ActDiff leads the listed search methods on all optimized metrics across three backbones (Appendix O). For IR+HPSv3 search, it also leads on held-out PickScore. With PickScore in the objective, HPSv2 is the sole reported held-out score and remains above the corresponding distilled baseline, although ActDiff is not uniformly best on HPSv2.

Trade-offs across objectives. Changing the objective changes which outputs are preferred by search. For ActDiff on FLUX, adding PickScore to the reported objective raises IR from 1.467 to 1.629 and PickScore from 22.853 to 22.909, while HPSv3 decreases from 15.024 to 13.856. SiD and SenseFlow exhibit the same direction of change in these three metrics (Table 2). HPSv2, which is excluded from both objectives, changes from 0.314 to 0.313 on FLUX, from 0.310 to 0.311 on SiD, and from 0.310 to 0.313 on SenseFlow. Thus adding a preference component redistributes performance, including among metrics that remain optimized. These are contrasts between complete configurations, including their reward normalization.

We examine whether preference and alignment rewards provide complementary signals within a prompt's candidate set. This observational study uses 3,200 images: four native samples for each of 200 prompts on four backbones. The prompts comprise 100 GenAI-Bench and 100 HPSv2 prompts; the backbones are FLUX.1-schnell, SD3.5-Large, SiD-SD3.5-Large, and SenseFlow-SD3.5-Large. The three scores are HPSv3, ImageReward, and CLIP-FlanT5 VQAScore (Ma et al., 2025b; Xu et al., 2023; Lin et al., 2024).

Complementarity within prompts. Table 3 reports within-prompt Pearson correlations. HPSv3–VQAScore correlation is 0.11 overall and -0.001 on SiD; the overall partial correlation controlling for ImageReward is 0.025. The participation ratio is 2.66 out of three dimensions (2.84 on SiD). The scorers capture complementary variation under the evaluated sampling conditions and are not interchangeable measures of image quality.

Implications for reward evaluation. Here, "orthogonality" denotes limited within-prompt agreement. The study motivates evaluating both preference and prompt alignment. The multi-reward benchmark above optimizes IR and HPSv3, optionally adding PickScore; VQAScore is used here to characterize complementarity. Appendix P gives the cohort, metric definitions, and interpretation limits.

5 Conclusion and Limitations

ActDiff exposes prompt conditioning and guidance as search actions, making reward-guided trajectory exploration available to few-step and deterministic diffusion samplers. Across three few-step backbones, ActDiff has the highest reported target-reward means among the listed methods under IR- and HPSv3-guided search. Multi-reward search balances preference criteria through explicit cross-metric trade-offs. The orthogonality study supports evaluating complementary preference and alignment signals together. The particle experiments extend the action interface beyond MCTS. Reward bias, rewrite fidelity, uncertainty, and unmeasured rewrite-generation and main-benchmark end-to-end costs remain limitations.

AI USE STATEMENT

Generative AI assisted manuscript editing, LaTeX formatting, citation checks, figure layout, notation consistency checks, proof exposition, sketching analysis pipelines, and preparation of analysis utilities. It also provided feedback on methodology and follow-up experimental design and assisted in interpreting and organizing reported results. The method uses a language model to propose prompt rewrites, as described in the experimental setup. We have reviewed all AI-assisted work. Reported numbers were checked against the underlying analysis outputs; figure construction is covered by automated provenance checks that confirm the original measurement streams are carried through unchanged; and citations were checked against the primary sources. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work evaluates test-time search over a frozen generator and does not release a new model or dataset. The human evaluation described in Section 4.2 and Appendix B.2 collected only preference judgments over generated images: method identities were concealed, image order was randomized, every question offered an explicit no-preference option, and no personally identifying information about the evaluators was collected or reported. Prompts are drawn from the public GenAI-Bench and HPSv2 benchmarks, and all images are model outputs rather than photographs of real people.

The method optimizes learned preference rewards, which are known to encode the biases of their training populations; Section 4.3 reports held-out metrics precisely so that gains on an optimized reward are not mistaken for broad quality improvements. Because search rewrites the conditioning prompt, outputs may drift from the user's literal request, and the prompt-bank construction does not guarantee semantic fidelity (Appendix K). Applied to a generator without safety filtering, reward-guided search could also amplify undesirable content; we did not evaluate that risk.

REPRODUCIBILITY STATEMENT

The appendix provides the extended search formulation (Appendix A), the proof of the reward-variation bound (Appendix F), MCTS pseudocode (Appendix D), trajectory diagnostics (Appendix H), baseline descriptions (Appendix L), and configuration tables (Appendix M and Tables 16 and 17). Prompt-bank construction is described in Appendix K, and benchmark and human-evaluation protocols are provided in Appendix B. Appendix P describes the reward-orthogonality cohort, scoring models, reported correlations, and interpretation limits. Appendix N.1 gives the separate SiD profiling measurements.

REFERENCES

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R. Devon Hjelm. Mutual information neural estimation. In *ICML*, 2018.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *ICLR*, 2024.

Black Forest Labs. FLUX.1 [schnell]. https://huggingface.co/black-forest-labs/FLUX.1-schnell, 2024. Fast variant of the FLUX.1 text-to-image model.

Yuzhu Chen, Fengxiang He, Shi Fu, Xinmei Tian, and Dacheng Tao. Adaptive time-stepping schedules for diffusion models. In *UAI*, 2024.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *ICML*, 2024.

Xingtong Ge, Xin Zhang, Tongda Xu, Yi Zhang, Xinjie Zhang, Yan Wang, and Jun Zhang. Senseflow: Scaling distribution matching for flow-based text-to-image distillation. In *ICLR*, 2026.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS Workshop on Deep Generative Models and Downstream Applications*, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

Vineet Jain, Kusha Sareen, Mohammad Pedramfar, and Siamak Ravanbakhsh. Diffusion tree sampling: Scalable inference-time alignment of diffusion models. In *NeurIPS*, 2025.

Jaihoon Kim, Taehoon Yoon, Jisung Hwang, and Minhyuk Sung. Inference-time scaling for flow models via stochastic generation and rollover budget forcing. In *NeurIPS*, 2025a.

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

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.

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.

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. *arXiv preprint arXiv:2406.13743*, 2024a. URL https://arxiv.org/abs/2406.13743.

Xiner Li, Yulai Zhao, Chenyu Wang, Gabriele Scalia, Gökcen Eraslan, Surag Nair, Tommaso Biancalani, Shuiwang Ji, Aviv Regev, Sergey Levine, and Masatoshi Uehara. Derivative-free guidance in continuous and discrete diffusion models with soft value-based decoding. In *NeurIPS Workshop on AI for New Drug Modalities*, 2024b.

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 *ICLR*, 2023.

Enshu Liu, Xuefei Ning, Huazhong Yang, and Yu Wang. A unified sampling framework for solver searching of diffusion probabilistic models. In *ICLR*, 2024.

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

Yuhang Ma, Xiaoshi Wu, Keqiang Sun, and Hongsheng Li. HPSv3: Towards wide-spectrum human preference score. In *ICCV*, 2025b.

Zanlin Ni, Yulin Wang, Yeguo Hua, Renping Zhou, Jiayi Guo, Jun Song, Bo Zheng, and Gao Huang. AdaGen: Learning adaptive policy for image synthesis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 48(3):2695–2713, 2026.

Yuta Oshima, Masahiro Suzuki, Yutaka Matsuo, and Hiroki Furuta. Inference-time text-to-video alignment with diffusion latent beam search. In *NeurIPS*, 2025.

Pinelopi Papalampidi, Olivia Wiles, Ira Ktena, Aleksandar Shtedritski, Emanuele Bugliarello, Ivana Kajic, Isabela Albuquerque, and Aida Nematzadeh. Dynamic classifier-free diffusion guidance via online feedback. In *ICLR*, 2026.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *ECCV*, 2024.

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.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. Learning to summarize from human feedback. In *NeurIPS*, 2020.

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 *CVPR*, 2024.

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.

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.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Frédo Durand, and Bill Freeman. Improved distribution matching distillation for fast image synthesis. In *NeurIPS*, 2024.

Mingyuan Zhou, Huangjie Zheng, Zhendong Wang, Mingzhang Yin, and Hai Huang. Score identity distillation: Exponentially fast distillation of pretrained diffusion models for one-step generation. In *ICML*, 2024.

Mingyuan Zhou, Yi Gu, Huangjie Zheng, Liangchen Song, Guande He, Yizhe Zhang, Wenze Hu, and Yinfei Yang. Score distillation of flow matching models. *arXiv preprint arXiv:2509.25127*, 2025.

A Extended Method Formulation

Most existing gradient-free trajectory-search methods use stochastic noise as their primary branching variable, either by varying the initial seed or by branching over intermediate stochastic innovations. Other work adapts inference schedules, solvers, or individual controls such as guidance strength, but typically through fixed policies or specialized adjustment mechanisms. ActDiff instead formulates multiple inference-time controls as a joint, per-step action space and makes this space available to general reward-guided search algorithms. The resulting framework can retain stochastic branching when available, while also supporting few-step and deterministic samplers where intermediate noise-based exploration is limited.

We use lowercase symbols for realized prompts, states, actions, and latents, and uppercase symbols for the corresponding random variables. The index $k \in \{0, \dots, K-1\}$ counts reverse transitions; t_k is the associated noise time. The original prompt c_0 stays fixed for reward evaluation.

Given an original prompt c_0, let $p_\theta(x_0 \mid c_0)$ denote the clean-image distribution induced by the default sampler. Reward-guided test-time inference seeks to bias generation toward outputs receiving higher reward under $r(x_0, c_0)$, with decoding implicit when x_0 is a latent representation. A convenient idealization of this preference is the reward-tilted distribution

$$p^\star_\lambda(x_0 \mid c_0) = \frac{p_\theta(x_0 \mid c_0) \exp(\lambda r(x_0, c_0))}{Z_\lambda(c_0)},$$

where $\lambda \geq 0$ controls the strength of the reward preference and $Z_\lambda(c_0)$ is the normalizing constant. We use $p^\star_\lambda$ as an idealized target that characterizes the desired reward-induced shift, rather than claiming that the procedures developed below produce exact samples from it.

Direct sampling from $p^\star_\lambda$ is generally intractable. Practical test-time methods instead generate candidate trajectories using the frozen generator, evaluate them through the reward, and allocate computation toward promising candidates. Although different search procedures allocate their budgets differently, the high-reward outputs they can discover depend fundamentally on the variables over which they are allowed to explore.

For a reverse-time schedule

$$T = t_0 > t_1 > \cdots > t_K = 0,$$

a broad class of diffusion samplers can be written as

$$x_{t_{k+1}} = F_{\theta,k}\left(x_{t_k}; c_0, \omega_0\right) + \sigma_{t_k} z_k, \qquad z_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I}),$$

where $F_{\theta,k}$ denotes the deterministic reverse update at stage k under the original prompt c_0 and default guidance scale ω_0, while $\sigma_{t_k} z_k$ denotes the stochastic innovation. For deterministic samplers, $\sigma_{t_k} = 0$.

Existing noise-based search methods construct alternative trajectories by varying the initial latent x_T or the intermediate innovations z_k, while keeping the inference-time controls (c_0, ω_0), and hence the reverse map $F_{\theta,k}$, fixed across branches.

Treating F_θ as fixed across branches is a choice of search space, rather than a requirement of the pretrained generator. In practice, the reverse update can be modified through inference-time controls without changing the model parameters. Standard sampling fixes these controls throughout generation; ActDiff instead allows the search procedure to select them.

A.1 Action Space Beyond Noise

Using stochastic noise as a proposal variable or branching action has been highly effective for reward-guided test-time sampling. However, its limitations become more apparent as efficient and deterministic samplers continue to emerge. In few-step distilled diffusion models, the short denoising horizon provides only a few opportunities for intermediate noise branching. In flow-matching models with ODE solvers, intermediate stochasticity is absent, so noise-based search largely reduces to initial-seed variation unless stochasticity is manually reintroduced. These settings motivate expanding the action space beyond noise. We therefore introduce ActDiff, a test-time optimization framework that uses inference-time controls as actions that directly modify the reverse transition.

ActDiff is designed as a general interface for searching inference-time controls that modify the reverse transition. In this work, we instantiate the framework using two complementary and widely exposed controls: prompt conditioning and classifier-free guidance scale. Both can be changed during sampling without updating the pretrained model parameters.

At each reverse stage, we distinguish two complementary action components: a *structured action*, which controls the deterministic reverse map, and a *noise action*, which selects the stochastic innovation. Together, they form the joint action applied at that stage.

At state $s_k = (x_{t_k}, t_k)$, we collect the selectable inference-time inputs into a structured action a_k and write the deterministic component of the controlled reverse update as

$$F_{\theta,k}(x_{t_k}, a_k).$$

To formalize the possible choices of a_k, we define the structured-control action space

$$\mathcal{A}_k(s_k; c_0) = \prod_{j=1}^{D_k} \mathcal{A}_{k,j}, \qquad \mathcal{A}_{k,j} = \left\{ a_{k,j}^{(1)}, \ldots, a_{k,j}^{(K_{k,j})} \right\}.$$

Here, D_k denotes the number of action coordinates; state and prompt dependence of the coordinate banks $\mathcal{A}_{k,j}$ is implicit. Each coordinate may specify an intermediate prompt rewrite, a guidance scale, a sampler operator, or another inference-time control.

The formulation above is intentionally general. For the concrete ActDiff instantiation studied in this paper, we set $D_k = 2$ and define

$$\mathcal{A}_{k,1} = \mathcal{C}(c_0), \qquad \mathcal{A}_{k,2} = \Omega_k(s_k),$$

so that

$$\mathcal{A}_k(s_k; c_0) = \mathcal{C}(c_0) \times \Omega_k(s_k).$$

Here, $\mathcal{C}(c_0)$ is a finite bank containing the original prompt and variants intended to preserve its meaning, while $\Omega_k(s_k) \subset \mathbb{R}_{>0}$ is a finite grid of CFG values. A structured action is therefore

$$a_k = (\widetilde{c}_k, \widetilde{\omega}_k).$$

The prompt coordinate changes the semantic conditioning supplied to the reverse process, while the CFG coordinate changes the strength of that conditioning.

At stage k, we additionally construct a finite set of stochastic branches,

$$\mathcal{Z}_k = \left(z_k^{(1)}, \ldots, z_k^{(M_k)} \right) \sim q_k(\cdot \mid s_k, c_0).$$

Here, q_k denotes the joint law of the available branches. We impose no independence assumption: the branches may be conditionally independent draws from the native reverse process or may be adaptively retained and expanded by a tree-based search procedure such as DTS (Jain et al., 2025).

Conditional on the realized branch set, the available action space is

$$\mathcal{B}_k(s_k; c_0, \mathcal{Z}_k) = \mathcal{A}_k(s_k; c_0) \times \left\{ z_k^{(1)}, \ldots, z_k^{(M_k)} \right\}.$$

We are now ready to define the policy and transition. The initial action-selection policy is uniform:

$$\pi_k^{(0)}(b \mid s_k, c_0, \mathcal{Z}_k) = \frac{1}{|\mathcal{B}_k(s_k; c_0, \mathcal{Z}_k)|}, \qquad b \in \mathcal{B}_k(s_k; c_0, \mathcal{Z}_k).$$

The selected action is $b_k = (a_k, z_k^\star) \sim \pi_k^{(0)}(\cdot \mid s_k, c_0, \mathcal{Z}_k)$, with $b_k \in \mathcal{B}_k(s_k; c_0, \mathcal{Z}_k)$.

The reverse sampler then produces the next latent according to

$$x_{t_{k+1}} = F_{\theta,k}(x_{t_k}, a_k) + \sigma_{t_k} z_k^\star,$$

where $F_{\theta,k}$ denotes the deterministic reverse update induced by the selected structured action, σ_{t_k} controls the stochasticity of the sampler, and $z_k^\star$ is the selected stochastic branch.

As is standard in diffusion search, intermediate rewards may be evaluated using the predicted clean sample $\widehat{x}_0^{(k)}$, with $\widehat{R}_k = r(\widehat{x}_0^{(k)}, c_0)$ for $k < K$. Sampling terminates at $t_K = 0$, with terminal reward $R = r(x_0, c_0)$.

For tree-based search, this policy is adaptive across rollouts. Let $\pi_k^{(n)}$ denote the action-selection policy at stage k during rollout n. It is initialized uniformly as above, while subsequent policies depend on the search statistics accumulated from previous rollouts. In particular, $b_k^{(n)} \sim \pi_k^{(n)}(\cdot \mid s_k, c_0, \mathcal{Z}_k)$.

After each rollout is evaluated, its rewards are propagated through the search tree, and the resulting visit and value statistics determine $\pi_k^{(n+1)}$. We leave the specific tree-policy update to the search algorithm.

The general formulation includes two common restricted search settings. For *structured-action search*, only one native noise realization is drawn at each stage, with $M_k = 1$ and $z_k^\star = z_k^{(1)} \overset{\text{ind}}{\sim} \mathcal{N}(0, I)$, and the policy searches only over $a_k \in \mathcal{A}_k(s_k; c_0)$. For *noise search*, the structured action is fixed at its default, $a_k = a_k^0$, and the policy selects among the candidate stochastic branches in $\mathcal{Z}_k$.

Furthermore, when the structured controls are fixed to their default values and only one Gaussian branch is sampled at each stage,

$$a_k = a_k^0, \qquad M_k = 1, \qquad z_k^\star = z_k^{(1)} \sim \mathcal{N}(0, I),$$

the action-selection policy is degenerate and the reverse update reduces to

$$x_{t_{k+1}} = F_{\theta,k}\big(x_{t_k}, a_k^0\big) + \sigma_{t_k} z_k^{(1)}.$$

Marginalizing the Gaussian noise induces the standard reverse kernel $x_{t_{k+1}} \sim p_{\theta,k}(\cdot \mid x_{t_k}, t_k, c_0)$. Identifying the next latent as the MDP action, $a_k^{\text{Diffusion-MDP}} := x_{t_{k+1}}$, recovers the standard diffusion-as-MDP formulation, in which the denoising model defines the policy over the next latent.

The next step is to construct an action space for each action dimension. The action space may be intrinsically discrete, such as a finite menu of prompts, tools, or adapter modules, or continuous, parameterized by a real-valued control such as the CFG scale or a guidance direction in embedding space. Both cases admit the same MDP treatment, but their search procedures differ: discrete spaces are naturally handled by tree expansion and categorical sampling, while continuous spaces require either discretization onto a finite grid or continuous-action search.

We illustrate the construction using prompt conditioning and guidance strength.

- **Prompt** (intrinsically discrete). We construct $\mathcal{C}(c_0)$ as a finite prompt bank containing c_0 together with $|\mathcal{C}| - 1$ variants intended to preserve its meaning.
- **CFG scale** (continuous). We discretize it onto a finite grid $\Omega_k(s_k) \subset \mathbb{R}_{>0}$ for compatibility with tree-search algorithms.

The joint space $\mathcal{C}(c_0) \times \Omega_k(s_k)$ is discrete, state-dependent, and finite—a structure any gradient-free search algorithm can optimize. We illustrate this construction in Fig. 1.

ActDiff searches the Cartesian product of these controls rather than optimizing prompt and guidance as isolated alternatives. Consequently, each action specifies a complete prompt–guidance configuration, allowing search to consider combinations that are unavailable when either coordinate is held fixed.

The joint action space is evaluated through the controlled ablations in Section B.4.

A.2 Information-Theoretic Perspective on the Action Space

The preceding subsection expands the search space from noise actions alone to joint actions that combine structured controls and stochastic innovations. We now provide an information-theoretic perspective on how the choice of action space constrains the variation in conditional expected reward available to search.

Let $C_0 \sim p_C$ denote the original prompt and let

$$\mathbf{B} = (B_0, \ldots, B_{K-1})$$

denote the complete joint-action trajectory. For the following analysis, fix a reference policy ρ and consider the induced process

$$C_0 \sim p_C, \qquad \mathbf{B} \sim \rho(\cdot \mid C_0), \qquad X_0 \sim p_\theta(\cdot \mid \mathbf{B}, C_0), \qquad R = r(X_0, C_0).$$

Define the conditional expected reward associated with a joint-action trajectory as

$$\mu_\rho(\mathbf{b}, c) \coloneqq \mathbb{E}\left[R \mid \mathbf{B} = \mathbf{b},\, C_0 = c\right].$$

We assume only that the final reward is uniformly bounded:

$$R \in [r_{\min}, r_{\max}] \qquad \text{almost surely.}$$

Consequently, $R \mid C_0 = c$ is V-sub-Gaussian for every c, with

$$V = \frac{(r_{\max} - r_{\min})^2}{4}.$$

Theorem A.1 (Information-limited reward variation)**.** *Under the reference distribution defined above,*

$$\mathbb{E}_{C_0 \sim p_C}\left[\mathrm{Var}_{\mathbf{B} \sim \rho(\cdot \mid C_0)}\left(\mu_\rho(\mathbf{B}, C_0)\right)\right] \le 2V\, I_{\theta,\rho}\left(\mathbf{B}; R \mid C_0\right). \tag{12}$$

Theorem A.1 gives a necessary condition for an action space to expose exploitable reward variation. The left-hand side of Equation 12 shows that variation in conditional expected reward is bounded by the reward information carried by the searched joint-action trajectory. If

$$I_{\theta,\rho}(\mathbf{B}; R \mid C_0)$$

is small, then the expected reward landscape must be nearly flat over the trajectories explored by ρ, leaving little reward variation for search to exploit. The converse does not necessarily hold: large mutual information permits greater reward variation but does not guarantee that a finite-budget search algorithm can identify and exploit it.

The proof follows from a variance–information inequality for sub-Gaussian random variables and is provided in Appendix F.

To empirically examine the reward dependence exposed by different action components, Fig. 2 reports the conditional mutual information

$$I(S; R \mid C_0),$$

where S denotes the control channel being evaluated. We consider noise, prompt, guidance, joint prompt–guidance, and combined noise–prompt–guidance channels under the same factorial reference distribution. In the evaluated setting, the joint prompt–guidance channel exhibits substantially greater reward dependence than noise alone, while combining structured and noise actions yields the largest estimated mutual information.

These estimates are consistent with Theorem A.1: expanding the action space can expose additional variation in conditional expected reward under the chosen reference distribution. The diagnostic is computed over factorial control configurations rather than complete adaptive search trajectories, and should therefore be interpreted as evidence about the action space rather than as a guarantee of finite-budget search improvement.

Having characterized the reward-relevant variation exposed by the joint-action space, we next describe how ActDiff instantiates an adaptive search policy π over this space using concrete search algorithms.

A.3 ActDiff Search with MCTS

ActDiff specifies which inference-time controls can be searched, while MCTS determines how the rollout budget is allocated among the resulting action trajectories. We use MCTS as our primary instantiation because its edges naturally represent discrete prompt–guidance choices and its value estimates allow information from completed generations to guide subsequent rollouts.

For one search instance, the original prompt c_0 and initial latent x_T are fixed. We use the fixed-noise restriction of the expanded-action formulation: at a given node, all candidate structured actions share the same stochastic branch, so $M_k = 1$. Consequently, selecting

$$b_k = (a_k, z_k^\star)$$

reduces to selecting its structured component

$$a_k = (\widetilde{c}_k, \widetilde{\omega}_k) \in \mathcal{A}_k(s_k; c_0).$$

For deterministic samplers, the stochastic term is absent altogether. A tree node represents the current denoising state $s_k = (x_{t_k}, t_k)$, and an outgoing edge represents the structured control applied in the next reverse transition.

Let $\pi_k^{(n)}$ denote the MCTS tree policy at stage k during rollout n. The selected action is

$$a_k^{(n)} \sim \pi_k^{(n)} \left(\cdot \mid s_k^{(n)}, c_0 \right).$$

The policy is adaptive across rollouts: $\pi_k^{(n)}$ depends on the tree statistics accumulated from rollouts $1, \ldots, n-1$.

Figure 1 illustrates candidate branches and terminal selection. In each rollout, starting from the root, the current tree policy follows previously expanded actions until it reaches a node with an unexpanded prompt–guidance choice. MCTS expands that action, applies the corresponding controlled reverse update, and adds the resulting state as a child. The remaining denoising stages are completed using the rollout policy π_{roll}. The resulting clean image $x_0^{(n)}$ receives terminal reward

$$R^{(n)} = r(x_0^{(n)}, c_0),$$

which is propagated through every state–action edge visited during the rollout.

Each edge stores its visit count $N(s, a)$ and the mean terminal reward of the rollouts that pass through it:

$$Q(s, a) = \frac{1}{N(s, a)} \sum_{n:\,(s,a) \in \mathcal{T}^{(n)}} R^{(n)},$$

where $\mathcal{T}^{(n)}$ denotes the state–action trajectory of rollout n.

Trajectory-feedback prior. We augment standard PUCT by replacing its fixed base prior P_0 with a state-dependent trajectory-feedback prior P_{tf}. Tree selection uses

$$a^\star = \arg \max_{a \in \mathcal{A}_k(s; c_0)} \left[Q(s, a) + c_{\text{puct}} P_{\text{tf}}(a \mid s) \frac{\sqrt{N(s)}}{1 + N(s, a)} \right], \tag{13}$$

where $Q(s, a)$ and $N(s, a)$ are the backed-up mean terminal reward and visit count, respectively.

When an action $a_k = (\widetilde{c}_k, \widetilde{\omega}_k)$ is expanded, we compute

$$u_k(s_k, a_k) = \text{RMS}\left(\widehat{x}_0^{\text{child}}(s_k, a_k) - \widehat{x}_0^{\text{base}}(s_k) \right), \tag{14}$$

$$d_k(s_k, a_k) = \text{RMS}\left(\epsilon_\theta(x_{t_k}, t_k, \widetilde{c}_k) - \epsilon_\theta(x_{t_k}, t_k, \varnothing) \right). \tag{15}$$

Here, u_k measures the action-induced trajectory deviation, while d_k measures the prompt-conditioned response. The prior P_{tf} reweights P_0 to favor informative conditioning responses while discouraging excessively large trajectory deviations. Unexpanded actions retain their relative base-prior weights, so these diagnostics affect only subsequent tree selection. The complete prior construction is given in Appendix H.

All other MCTS operations remain unchanged: terminal rewards update Q and N through standard backup. The method requires only forward denoising calls and terminal reward queries and therefore remains applicable to black-box rewards, few-step models, and deterministic samplers.

A.4 ActDiff with Particle-Based Search

ActDiff can also be incorporated into particle-based search methods such as Sequential Monte Carlo (SMC). Standard SMC alternates between propagating a population of P denoising trajectories and reallocating particles toward high-reward regions through weighting and resampling. ActDiff retains this procedure but expands the set of transitions available during propagation.

At denoising stage k, particle i contains a latent $X_{t_k}^{(i)}$ together with the structured actions selected along its trajectory. Before propagation, the particle samples a prompt–CFG action according to

$$A_k^{(i)} \sim \pi_k \left(\cdot \mid S_k^{(i)}, C_0 \right), \qquad S_k^{(i)} = \left(X_{t_k}^{(i)}, t_k \right).$$

It then advances through the corresponding controlled reverse transition,

$$X_{t_{k+1}}^{(i)} = F_{\theta,k}\left(X_{t_k}^{(i)}, A_k^{(i)}\right) + \sigma_{t_k} Z_k^{(i)}.$$

Thus, different particles may explore different prompt and guidance choices in addition to their native stochastic variation. For deterministic samplers, the stochastic term is absent, but the structured actions still produce distinct particle trajectories.

After propagation, each particle is evaluated using the reward potential of the underlying particle method. Let $G_{k+1}^{(i)}$ denote this incremental potential. The particle weights are updated as

$$\widetilde{w}_{k+1}^{(i)} = w_k^{(i)} G_{k+1}^{(i)}, \qquad w_{k+1}^{(i)} = \frac{\widetilde{w}_{k+1}^{(i)}}{\sum_{j=1}^{P} \widetilde{w}_{k+1}^{(j)}}.$$

The population is then resampled according to $\{w_{k+1}^{(i)}\}_{i=1}^{P}$.

Resampling acts on complete particle trajectories rather than on controls in isolation. Consequently, a structured action that produces a high-weight particle is inherited by more descendants, while actions associated with low-weight trajectories receive less subsequent computation. In this way, the particle population gradually concentrates on promising structured-control trajectories.

ActDiff preserves the overall propagate–weight–resample structure of the underlying particle method while augmenting the set of transitions available during propagation. Instead of propagating every particle under the default structured action

$$a_k^0 = (c_0, \omega_0),$$

particle i may select

$$a_k^{(i)} \in \mathcal{A}_k(s_k^{(i)}; c_0) = \mathcal{C}(c_0) \times \Omega_k(s_k^{(i)})$$

before applying the corresponding controlled reverse transition. The resulting particles are then evaluated, weighted, and resampled using the machinery of the underlying particle method; any proposal-dependent correction remains specific to that method.

Thus, ActDiff acts at the level of the action space: it provides particle-based search methods with structured prompt–guidance transitions in addition to their native stochastic variation, without prescribing a new reward potential or resampling rule. Initial seeds provide coverage across generation modes; structured actions add refinement within each seed, including in few-step or deterministic samplers with limited stochastic branching.

B Full Benchmark Results and Evaluation Protocols

Setup. For multi-step generation, we use Stable Diffusion 3.5 Large (Esser et al., 2024) as the base model. Its native ODE sampler lacks the stochastic branching required by some noise-based search baselines. Following Kim et al. (2025a), we inject standard Gaussian noise into the reverse process and label this modified version "SDE sampler" to enable comparisons with those methods. For few-step generation, we evaluate four-step SiD-SD3.5-Large (Zhou et al., 2025), SenseFlow-SD3.5-Large (Ge et al., 2026), and FLUX.1-schnell (Black Forest Labs, 2024), representing recent distillation families such as SiD (Zhou et al., 2024), DMD (Yin et al., 2024), and adversarial or accelerated distillation (Sauer et al., 2024). Following the evaluation setting of Jain et al. (2025), we use a 200-prompt subset of the HPSv2 test benchmark (Wu et al., 2023).

Evaluation metrics. We use human-preference reward models to assess visual quality and alignment with the original prompt c_0: ImageReward (IR) (Xu et al., 2023), a BLIP-based model of image–text alignment and visual quality; HPSv2 (Wu et al., 2023), a CLIP-based preference model; HPSv3 (Ma et al., 2025b), a VLM-based preference model; and PickScore (Kirstain et al., 2023), a CLIP-based model trained to predict human preferences.

Baselines. We compare ActDiff with the base models and existing test-time scaling (TTS) methods, grouped by whether they search from one initial noise seed or across multiple seeds. The single-root configurations include Search over Paths (SoP) (Ma et al., 2025a), a genetic algorithm (GA), and Diffusion Tree Sampling (DTS) (Jain et al., 2025). Multi-root methods explore pools of trajectories initialized with different noise seeds. They include Diffusion Alignment as Sampling (DAS) (Kim et al., 2025b), Feynman–Kac steering (FK-Steering) (Singhal et al., 2025), beam search (Oshima et al., 2025), and best-of-N (BoN) (Stiennon et al., 2020). For DAS (Kim et al., 2025b) and FK-Steering (Singhal et al., 2025), we use reward-gradient proposals where computationally feasible and applicable gradient-free variants otherwise. We report each method's backbone NFE budget alongside its scores. The tables include both nearby-budget comparisons and higher-budget baselines; Figure 4 shows performance over a range of budgets.

Implementation details. Following Jain et al. (2025), we discretize the bounded, continuous CFG action space into per-step dynamic banks. We use Qwen3-4B (Yang et al., 2025) as a lightweight prompt rewriter for the benchmark. The benchmark experiments are implemented in PyTorch (Paszke et al., 2019) and run on a single cluster node with up to eight NVIDIA A100 GPUs. Model-specific configurations and search hyperparameters are provided in Appendix M.

B.1 Quantitative Experiments

Few-step distilled models. Under ImageReward-guided search, ActDiff has the highest reported IR means among the listed methods on FLUX.1-schnell (1.687), SiD-SD3.5 (1.603), and SenseFlow-SD3.5 (1.640; Table 4). The same table reports HPSv2, HPSv3, and PickScore to show how these IR gains relate to metrics excluded from search.

Table 5 reports performance when guided by HPSv3, a stronger VLM-based reward. ActDiff also achieves superior performance over existing search baselines on the target HPSv3 metric, reaching 15.528 on FLUX.1-schnell, 13.750 on SiD-SD3.5, and 14.939 on SenseFlow-SD3.5. ActDiff does not require a differentiable reward. DAS and FK-Steering also support gradient-free variants, but the cost of differentiating large VLM-based rewards can limit the use of their reward-gradient proposals.

Table 4: ImageReward-guided search across FLUX.1-schnell (Black Forest Labs, 2024), SiD-SD3.5 (Zhou et al., 2025), and SenseFlow-SD3.5 (Ge et al., 2026). Models are grouped across columns; higher reward scores are better.

	FLUX.1-schnell					SiD-SD3.5					SenseFlow-SD3.5				
Method	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick
Baselines															
Multi-step Baseline	56	1.269	0.308	13.613	22.784	56	1.022	0.308	12.240	22.807	56	1.022	0.308	12.240	22.807
Distilled Baseline	4	1.163	0.304	13.558	22.735	4	1.180	0.305	12.340	22.756	4	1.269	0.304	12.220	22.652
Multiple-root search															
DAS*	256	1.488	0.312	13.667	**22.916**	256	1.518	0.318	12.460	22.799	256	1.488	0.308	12.511	22.655
+ ActDiff		1.503	0.313	13.703	22.882		1.529	0.318	12.514	22.833		1.525	0.309	12.560	22.702
FK-Steering*	256	1.515	**0.318**	13.658	22.903	256	1.479	0.318	12.521	22.884	256	1.527	0.309	12.488	**22.920**
+ ActDiff		1.539	0.312	**14.234**	22.530		1.519	0.318	12.528	22.901		1.539	**0.310**	**12.778**	22.810
BoN	128	1.583	0.314	13.702	22.904	64	1.513	0.313	12.496	22.736	64	1.544	0.309	12.360	22.720
Beam	455	1.557	0.314	13.600	22.785	455	1.462	0.311	12.353	22.603	455	1.349	0.304	12.299	22.607
Single-root search															
SoP	128	1.446	0.312	13.632	22.725	128	1.470	0.301	12.422	22.219	128	1.343	0.306	12.315	22.650
+ ActDiff		1.464	0.304	13.439	22.702		1.406	0.304	12.333	22.206		1.385	0.304	12.122	22.593
GA	768	1.537	0.313	13.601	22.792	768	1.379	0.306	12.257	22.385	768	1.350	0.305	12.300	22.660
DTS	108	1.251	0.301	13.593	22.762	108	1.245	0.306	12.263	22.767	109	1.281	0.304	12.235	22.840
DTS*	108	1.363	0.305	13.648	22.802	108	1.490	0.312	**12.555**	22.843	109	1.476	0.309	12.406	22.900
Dynamic CFG†	8	1.402	0.307	13.625	22.752	8	1.236	0.304	12.307	22.746	8	1.346	0.308	12.271	22.800
ActDiff	136	**1.687**	0.316	13.713	22.903	138.3	**1.603**	**0.320**	12.540	**22.927**	139.1	**1.640**	**0.310**	12.471	22.883

Pick denotes PickScore. *Note*: Bold reward scores mark column maxima among search methods; ties are bold. Multiple-root methods operate over multiple initial roots or candidate trajectories, while single-root methods expand the search from a fixed initial root. For Dynamic CFG, the action space is restricted to CFG selection only. *DAS/FK-Steering use reward gradients where feasible and gradient-free variants otherwise. †Dynamic CFG searches guidance only.

Table 5: HPSv3-guided search across FLUX.1-schnell (Black Forest Labs, 2024), SiD-SD3.5 (Zhou et al., 2025), and SenseFlow-SD3.5 (Ge et al., 2026). Models are grouped across columns; higher reward scores are better.

	FLUX.1-schnell					SiD-SD3.5					SenseFlow-SD3.5				
Method	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick
Baselines															
Multi-step Baseline	56	1.269	0.308	13.613	22.784	56	1.022	0.308	12.240	22.807	56	1.022	0.308	12.240	22.807
Distilled Baseline	**4**	1.213	0.308	13.558	22.801	**4**	1.180	0.308	12.340	22.756	**4**	1.269	0.308	12.220	22.652
Multiple-root search															
DAS*	✗	✗	✗	✗	✗	256	1.134	0.309	12.980	**22.902**	256	1.285	0.308	14.030	**22.803**
FK-Steering*	256	1.263	0.311	14.990	22.833	256	1.276	**0.312**	13.490	22.860	256	**1.346**	0.311	13.680	22.800
BoN	256	1.304	0.309	14.732	**22.903**	256	1.243	0.311	13.003	22.873	256	1.310	0.310	13.990	22.721
Beam	455	1.325	0.307	14.836	22.872	455	1.255	0.306	12.850	22.853	455	1.260	0.308	13.320	22.747
Single-root search															
SoP	60	1.234	0.310	14.028	22.851	60	1.087	0.309	12.670	22.792	60	1.284	0.307	12.650	22.733
GA	768	1.256	0.307	14.218	22.832	768	1.102	0.306	12.730	22.721	768	1.297	0.305	12.910	22.703
DTS	256	1.312	0.302	13.490	22.843	256	1.224	0.303	13.644	22.718	256	1.236	0.304	12.023	22.705
DTS*	256	1.285	0.311	14.670	22.851	256	1.272	0.310	12.833	22.854	256	1.243	0.305	13.885	22.802
Dynamic CFG†	8	1.241	0.304	13.803	22.824	8	1.164	0.306	12.380	22.773	8	1.290	0.305	12.470	22.708
ActDiff	240	**1.341**	**0.312**	**15.528**	22.863	246.7	**1.281**	0.310	**13.750**	22.864	248.4	1.307	**0.312**	**14.939**	22.793

Pick denotes PickScore. *Note*: Bold reward scores mark column maxima among search methods; ties are bold. ✗ indicates that the method is computationally impractical under the evaluated budget. For FK-Steering, we use the no-reward-gradient variant for FLUX.1-schnell. For Dynamic CFG, the action space is restricted to CFG selection only. †Dynamic CFG searches guidance only.

Table 6: ImageReward-guided search on SD3.5-Large. Configured NFEs follow Table 17.

	SD3.5-Large (Esser et al., 2024)			
Method	Cfg. NFE	IR↑	HPSv2↑	PickScore↑
Baseline	**28**	1.022	0.308	22.807
BoN	448	1.266	0.310	22.914
SoP	✗	✗	✗	✗
DAS	448	1.467	0.314	23.054
FK-Steering	224	1.474	0.314	23.060
Beam	3052	1.324	0.301	22.614
DTS (ODE Sampler)	✗	✗	✗	✗
DTS* (ODE Sampler)	✗	✗	✗	✗
ActDiff	≈ 643 (≤ 980)	**1.660**	**0.316**	**23.128**

Table 7: HPSv3-guided search on SD3.5-Large. Configured NFEs follow Table 17.

	SD3.5-Large (Esser et al., 2024)			
Method	Cfg. NFE	IR↑	HPSv3↑	PickScore↑
Baseline	**28**	1.022	11.717	22.807
BoN	448	1.211	13.789	22.845
SoP (ODE Sampler)	✗	✗	✗	✗
SoP (SDE Sampler)	≈ 408	1.189	12.308	22.846
DAS	448	1.277	13.991	22.932
FK-Steering	224	1.304	14.026	22.959
Beam	3052	1.319	13.560	22.813
DTS (ODE Sampler)	✗	✗	✗	✗
DTS* (ODE Sampler)	✗	✗	✗	✗
ActDiff	≈ 643 (≤ 980)	1.350	14.265	22.990

Note: Configured NFEs count batched denoiser invocations per trajectory step, with CFG branches batched together; reward-model evaluations are excluded. Approximate values are configured means, and the parenthesized ActDiff value is the stated worst-case bound. ✗ indicates that the specific method requires intermediate noise branching during the reverse process, which is incompatible with ODE-based samplers.

Table 8: ActDiff with particle-based search under single- and multi-reward objectives. The +ActDiff variants add prompt–guidance actions while retaining the underlying weighting and resampling procedure. Bold reward scores compare each native method with its +ActDiff variant; ties are bold.

ImageReward-guided search

	FLUX.1-schnell					SiD-SD3.5					SenseFlow-SD3.5				
Method	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick
DAS	256	1.555	0.305	13.762	**22.807**	256	1.518	**0.318**	12.460	22.799	256	1.488	0.308	12.511	22.655
+ActDiff	256	**1.563**	**0.310**	**14.203**	22.682	256	**1.529**	**0.318**	**12.514**	**22.833**	256	**1.525**	**0.309**	**12.560**	**22.702**
FK-Steering	256	1.515	**0.318**	13.658	**22.903**	256	1.479	**0.318**	12.521	22.884	256	1.527	0.309	12.488	**22.920**
+ActDiff	256	**1.539**	0.312	**14.234**	22.530	256	**1.519**	**0.318**	**12.528**	**22.901**	256	**1.539**	**0.310**	**12.778**	22.810

HPSv3-guided search

	FLUX.1-schnell					SiD-SD3.5					SenseFlow-SD3.5				
Method	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick
DAS	256	1.273	0.305	14.166	**22.807**	256	1.238	**0.318**	13.901	22.799	256	**1.261**	0.308	13.885	22.655
+ActDiff	256	**1.287**	**0.310**	**14.593**	22.682	256	**1.255**	**0.318**	**14.139**	**22.833**	256	1.232	**0.309**	**14.162**	**22.702**
FK-Steering	256	**1.389**	**0.312**	14.210	**22.831**	256	1.205	**0.318**	13.727	22.884	256	1.205	0.309	13.583	**22.920**
+ActDiff	256	1.323	**0.312**	**14.624**	22.630	256	**1.212**	**0.318**	**14.125**	**22.901**	256	**1.219**	**0.310**	**13.801**	22.810

ImageReward + HPSv3

	FLUX.1-schnell					SiD-SD3.5					SenseFlow-SD3.5				
Method	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick
DAS	256	1.404	0.305	14.636	**22.807**	256	1.372	**0.318**	13.841	22.799	256	1.386	**0.308**	13.749	22.655
+ActDiff	256	**1.451**	**0.307**	**14.829**	22.762	256	**1.448**	0.316	**13.931**	**22.812**	256	**1.406**	**0.308**	**13.921**	**22.702**
FK-Steering	256	1.447	**0.313**	14.592	22.820	256	1.366	0.317	13.320	22.884	256	1.424	0.314	13.546	**22.817**
+ActDiff	256	**1.463**	0.312	**14.773**	**22.855**	256	**1.415**	**0.318**	**13.845**	**22.897**	256	**1.443**	**0.315**	**13.916**	22.787

ImageReward + HPSv3 + PickScore

	FLUX.1-schnell					SiD-SD3.5					SenseFlow-SD3.5				
Method	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick
DAS	256	1.432	0.310	13.837	22.863	256	1.418	**0.309**	13.480	22.798	256	1.436	0.310	13.566	22.810
+ActDiff	256	**1.461**	**0.313**	**14.200**	**22.983**	256	**1.479**	**0.309**	**13.514**	**22.833**	256	**1.441**	**0.311**	**13.973**	**22.869**
FK-Steering	256	1.440	0.311	13.726	22.844	256	1.437	**0.309**	13.274	**22.818**	256	1.421	**0.310**	13.193	22.813
+ActDiff	256	**1.476**	**0.315**	**13.950**	**22.950**	256	**1.455**	0.308	**13.482**	22.550	256	**1.430**	**0.310**	**13.435**	**22.834**

Pick denotes PickScore. All metrics use the original evaluation prompt. Higher reward scores are better. All configurations use 256 NFEs.

Negative transfer from action augmentation. Table 4 also contains regressions. On SiD, adding ActDiff to SoP lowers IR by 0.064 (1.470 to 1.406), HPSv3 by 0.089, and PickScore by 0.013, while HPSv2 rises by 0.003. On SenseFlow, SoP's IR improves by 0.042, while HPSv2, HPSv3, and PickScore decline by 0.002, 0.193, and 0.057, respectively. The enlarged action space can change how a finite search budget is allocated and which metric trade-offs are selected; these explanations remain hypotheses. The results support action-space compatibility, with gains depending on the optimizer, backbone, and metric.

Multi-step diffusion model. ActDiff also applies to ODE-based flow-matching models without introducing an SDE sampler. Table 6 evaluates ImageReward-guided search on SD3.5-Large, and Table 7 reports HPSv3-guided search. Intermediate noise branching is unavailable in an ODE sampler, although initial-seed search remains possible. Structured prompt and guidance actions provide additional choices along the trajectory. ActDiff achieves the highest reported target-reward means among the listed methods under both objectives.

Multi-reward guided sampling. We evaluate two distinct preference composites: ImageReward (IR) plus HPSv3, and IR plus HPSv3 plus PickScore. The IR+HPSv3 comparison (Table 20) gives ActDiff the highest IR, HPSv3, and PickScore means among the listed search methods on all three few-step backbones. PickScore and HPSv2 are excluded from this objective. Under the IR+HPSv3+PickScore objective (Table 21), ActDiff again leads on the three optimized metrics. HPSv2, the only held-out metric reported for this objective, is above the distilled baseline on all three backbones, although ActDiff does not lead every HPSv2 column. These comparisons support joint reward optimization. The separate orthogonality study in Section 4.3 examines agreement between preference and alignment scores on native samples.

B.1.1 Scaling with inference budget

We further report how ActDiff scales against baselines as a function of NFEs in Fig. 4. The results are best read along two largely independent axes of test-time variation: *global* diversity from the choice of initial noise seed, and *within-trajectory* refinement from the structured prompt–guidance action space. Multi-root baselines spend their budget on the first axis, whereas ActDiff, searching from a fixed seed, spends it on the second.

This decomposition explains the budget-dependent behavior. At small budgets (e.g., below 100 NFEs), maintaining several initial-noise seeds is a cheap and effective source of variation, so multi-root baselines occupy a sweet spot and ActDiff performs on par with, rather than ahead of, these baselines. As the budget grows, the marginal value of additional seeds diminishes, while structured action search keeps surfacing reward-relevant trajectory variations; ActDiff thus catches up and then pulls ahead by expanding the prompt–guidance action space.

Because the two axes are complementary, they can be combined. In the last two grids for FLUX.1-schnell, SiD-SD3.5, and SenseFlow-SD3.5, and the last grid for SD3.5-Large, we reintroduce the initial seed as an additional search dimension for ActDiff: we maintain two initial-noise seeds and select the best result after action search. This joint search over global seed-level diversity and structured denoising actions yields the strongest configuration, confirming that action-space refinement is orthogonal to—and composable with—seed-level exploration.

B.2 Human Preference Evaluation

We conduct a pairwise human evaluation on GenAI-Bench (Li et al., 2024a) to measure overall preference, original-prompt alignment, and visual quality. GenAI-Bench contains compositionally challenging prompts spanning diverse objects, attributes, counts, actions, and spatial relationships, making it suitable for assessing both perceptual quality and faithfulness to the original conditioning prompt.

We evaluate four diffusion-model backbones: FLUX.1-schnell, SiD-SD3.5, SD3.5-Large, and SenseFlow-SD3.5-Large. For each backbone, we select 50 GenAI-Bench prompts and use the same prompts for all compared methods. ActDiff is independently compared against direct baseline sampling, Best-of-N (BoN), and Diffusion Alignment as Sampling (DAS). For each prompt–model–comparator combination, we construct one pair containing the ActDiff output and the corresponding

comparator output. This produces

$$4 \text{ models} \times 50 \text{ prompts per model} \times 3 \text{ comparators} = 600$$

unique image pairs.

The 600 pairs are distributed approximately evenly among eight human evaluators, with each pair assigned to one evaluator and each evaluator completing approximately 75 comparisons. Evaluators are shown the original GenAI-Bench prompt and two generated images, while the method identities are concealed and the left–right presentation order is randomized. For each pair, evaluators answer three questions: (Q1) which image they prefer overall, (Q2) which image better aligns with the original prompt, and (Q3) which image has higher visual quality. For each question, evaluators may select the ActDiff output, the comparator output, or indicate no preference.

Missing or invalid responses are excluded from the corresponding analysis, whereas no-preference responses are retained as ties and assigned a score of 0.5. For each model, comparator, and evaluation question, we calculate the ActDiff preference rate as

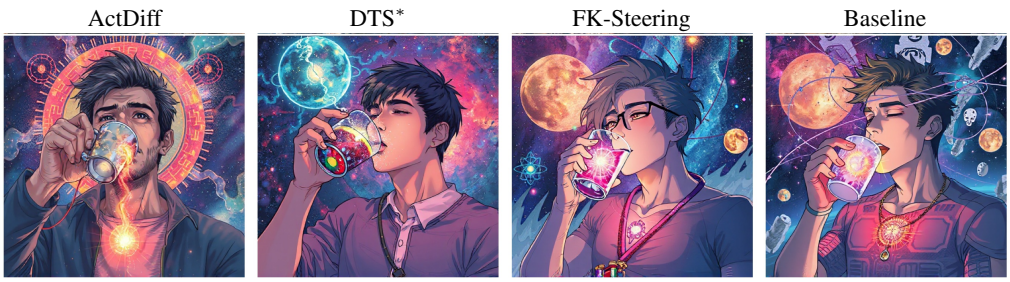
$$\text{PreferenceRate} = \frac{W + 0.5N_{\text{tie}}}{W + L + N_{\text{tie}}} \times 100\%,$$

where W, L, and N_{tie} denote the numbers of ActDiff wins, losses, and ties, respectively. Consequently, the effective number of valid judgments may be smaller than the 50 attempted comparisons in some settings.

Prompt–image alignment evaluation. We use VQAScore with CLIP-FlanT5-XXL to assess prompt–image alignment (Lin et al., 2024). It measures the probability of answering "Yes" to a question asking whether the image depicts the original prompt; it is not a hard binary correctness score or an average of separately generated condition-level questions. Appendix P.2 describes its use in measuring reward complementarity. A score included in a search objective is an optimized metric rather than a held-out evaluator.

B.3 Qualitative Experiments

Figures 6, 7, and 8 compare ActDiff with DTS*, FK-Steering, and baseline sampling under ImageReward- and HPSv3-guided search on FLUX.1-schnell, SiD-SD3.5, and SenseFlow-SD3.5. Each comparison uses matched compute budgets and the same random seeds.



(a) HPSv3-guided search, second prompt; the first is Figure 3.



(b) ImageReward-guided search, second prompt; the first is Figure 3.

Figure 6: Qualitative comparison of ActDiff with representative baselines on FLUX.1-schnell using matched compute budgets and the same random seeds.



(a) HPSv3-guided search.



(b) ImageReward-guided search, second prompt; the first is Figure 3.

Figure 7: Qualitative comparison of ActDiff with representative baselines on SiD-SD3.5-Large using matched compute budgets and the same random seeds.



(a) HPSv3-guided search, second prompt; the first is Figure 3.



(b) ImageReward-guided search.

Figure 8: Qualitative comparison of ActDiff with representative baselines on SenseFlow-SD3.5-Large using matched compute budgets and the same random seeds.

B.4 Ablation Study

We study action-bank sizes, exploration strength, prior variants, and prompt-rewriter capacity on SiD-SD3.5 using ImageReward. Larger banks improve the optimized score in the reported settings (Table 9), while held-out metrics need not improve monotonically. The exploration-constant comparison is in Table 10; the prior comparison is in Table 13. Qwen3-8B improves IR and HPSv2 relative to Qwen3-4B, while Qwen3-4B has the higher PickScore (Table 11). The SD3.5 search-space ablation (Table 12) compares joint search with each individual control removed.

Contribution of the action coordinates. Tables 12 and 9 report the SD3.5 control ablation and the SiD bank-size study, respectively. On SD3.5, joint search has higher IR, HPSv2, and PickScore than either restricted action space. The SiD results show why bank size should be assessed with several metrics: increasing the guidance bank from seven to nine values raises IR while lowering HPSv2, and increasing the prompt bank from four to seven variants raises IR and HPSv2 while slightly lowering PickScore. These results support the usefulness of both controls.

Table 9: Bank sizes on SiD-SD3.5.

Setting	Size	Metric		
		ImageReward ↑	HPSv2 ↑	PickScore ↑
CFG Bank Size	3	1.433	0.301	22.730
	7	1.530	**0.308**	22.810
	9	**1.590**	0.305	**22.830**
Prompt Bank Size	1	1.407	0.302	22.860
	4	1.576	0.310	**22.870**
	7	**1.595**	**0.312**	22.850

Table 10: Exploration constant on SiD-SD3.5.

c_{puct}	ImageReward ↑	HPSv2 ↑	PickScore ↑
0.5	1.573	0.308	22.830
1.0	1.587	0.308	22.850
1.41	**1.604**	**0.309**	22.840
2.0	1.600	0.308	**22.860**

Table 11: Prompt-rewriter comparison on SiD-SD3.5.

Model	ImageReward ↑	HPSv2 ↑	PickScore ↑
Qwen3-4B	1.609	0.312	**22.985**
Qwen3-8B	**1.617**	**0.315**	22.974

Table 12: Search-space ablation on SD3.5. Removing CFG search fixes the guidance scale to the default value; removing prompt search disables prompt rewriting.

Method	IR↑	HPSv2↑	PickScore↑
ActDiff w/o guidance search	1.493	0.317	22.620
ActDiff w/o prompt search	1.487	0.313	22.320
ActDiff	**1.649**	**0.321**	**22.953**

Table 13: Comparison of prior variants on SiD-SD3.5.

Method	ImageReward↑	HPSv2↑	PickScore↑
Prior	**1.598**	**0.313**	**22.812**
No Prior	1.549	0.312	22.773
Only d	1.565	0.312	22.782
Only u	1.571	0.313	22.761
Uniform Prior	1.547	0.311	22.767
Random Prior	1.547	0.304	22.732

B.5 Action and Reward Ablations in SMC-Based Search

What is varied. Table 8 crosses two particle methods (DAS and FK-Steering), three backbones, and four reward objectives. Within each method–backbone–objective block, the comparison is native propagation versus propagation augmented with prompt–guidance actions. The native weighting and resampling machinery is retained. The two composite-objective blocks are summarized as signed differences in Table 14; all original scores remain in the full table. This tests applicability to particle search separately from the MCTS control ablations in Tables 12 and 9.

Optimized and held-out scores. For IR+HPSv3, all 12 optimized-score differences are positive across the six method–backbone pairs. Of the 12 held-out differences (PickScore and HPSv2), seven are positive, one is zero at the displayed precision, and four are negative. For IR+HPSv3+PickScore, 17 of 18 optimized-score differences are positive; SiD FK-Steering's PickScore decreases by 0.268.

Table 14: Action-augmentation contrasts in particle search: Δ = (+ActDiff) − native method, computed from Table 8. These are differences of reported means within each objective, with 256 NFEs stated for every source row. Positive values favor +ActDiff.

Method	Metric	FLUX.1-schnell	SiD-SD3.5	SenseFlow-SD3.5
IR + HPSv3				
DAS	IR	+0.047	+0.076	+0.020
	HPSv3	+0.193	+0.090	+0.172
	PickScore†	-0.045	+0.013	+0.047
	HPSv2†	+0.002	-0.002	+0.000
FK-Steering	IR	+0.016	+0.049	+0.019
	HPSv3	+0.181	+0.525	+0.370
	PickScore†	+0.035	+0.013	-0.030
	HPSv2†	-0.001	+0.001	+0.001
IR + HPSv3 + PickScore				
DAS	IR	+0.029	+0.061	+0.005
	HPSv3	+0.363	+0.034	+0.407
	PickScore	+0.120	+0.035	+0.059
	HPSv2†	+0.003	+0.000	+0.001
FK-Steering	IR	+0.036	+0.018	+0.009
	HPSv3	+0.224	+0.208	+0.242
	PickScore	+0.106	-0.268	+0.021
	HPSv2†	+0.004	-0.001	+0.000

†Held out from the corresponding objective. Each block compares native and augmented search under the same named objective; the blocks are not a controlled estimate of adding a reward component. In particular, PickScore becomes optimized in the second block. No significance test is computed from these aggregate means.

Its held-out HPSv2 also decreases by 0.001, while the remaining five HPSv2 differences are positive or unchanged. These are counts of metric cells in the supplied tables, not independent trials.

What the comparisons isolate. The native/+ActDiff contrast evaluates the complete action augmentation. It does not separate prompt actions from guidance actions within SMC, or step-wise action changes from selecting one fixed action per particle. Moreover, comparisons across objective blocks can also reflect changes in reward normalization or potential strength. Thus the current SMC results support the action interface's applicability and describe its trade-offs; component-wise SMC controls require additional runs.

C Additional Method Details

This appendix provides additional theoretical and implementation details omitted from the main text. Appendix D gives the ActDiff-MCTS pseudocode. Appendix E defines the information quantities, and Appendix F proves Eq. 12. Appendix G further discusses the limitations of noise-centric search in few-step and deterministic samplers. Appendix H details the state-dependent diagnostics and trajectory-feedback prior used in ActDiff-MCTS. Appendix I describes the mutual-information diagnostic used in our empirical analysis. Appendix A.4 explains how the same structured action space can be incorporated into particle-based search methods.

D Detailed MCTS Algorithm

Algorithm 1 describes ActDiff-MCTS. For compactness, $\mathcal{A}(s)$ abbreviates $\mathcal{A}_k(s_k; c_0)$ and $\Omega(s)$ abbreviates $\Omega_k(s_k)$ at $s = s_k$.

Algorithm 1 ActDiff-MCTS

Input: initial latent x_T, original prompt c_0, prompt bank $\mathcal{C}(c_0)$, root guidance bank $\Omega_0(s_0)$, simulation budget N_{sim}
State: $s = s_k = (x_{t_k}, t_k)$
Action: $a = (\widetilde{c}, \widetilde{\omega}) \in \mathcal{A}(s)$
Statistics: $N(s)$, $N(s,a)$, $Q(s,a)$, $P_{\text{tf}}(a \mid s)$
Initialize root node $s_0 = (x_T, T)$
for $n = 1$ to N_{sim} **do**
  $s \leftarrow s_0,\ \mathcal{P} \leftarrow \emptyset$
  Selection:
  while s is expanded, nonterminal, and all actions in $\mathcal{A}(s)$ have been tried **do**
    compute $P_{\text{tf}}(a \mid s)$ from the base prior and observed diagnostics (Appendix H)
    select

$$a^* = \arg\max_a \left[Q(s,a) + c_{\text{puct}}\, P_{\text{tf}}(a \mid s) \frac{\sqrt{N(s)}}{1 + N(s,a)} \right]$$

    append (s, a^*) to $\mathcal{P}$ and move to the corresponding child
  end while
  Expansion:
  if s is nonterminal and $\mathcal{A}(s)$ is not initialized **then**
    build the admissible CFG bank $\Omega(s)$ and define

$$\mathcal{A}(s) = \mathcal{C}(c_0) \times \Omega(s)$$

  end if
  choose one unexpanded action $a \in \mathcal{A}(s)$ using the prior
  apply a for one step, create child node s'
  append (s, a) to $\mathcal{P}$ and set $s \leftarrow s'$
  Rollout:
  complete the remaining denoising steps and compute terminal reward R
  Backup:
  for each (s, a) in $\mathcal{P}$ **do**
    update $N(s)$, $N(s,a)$, and $Q(s,a)$ using reward R
  end for
end for
return the highest-reward terminal image found

E Information-Theoretic Analysis

We analyze why the searched variable matters in reward-guided test-time search. In this section the original prompt c_0 is fixed, and we suppress it in distribution and reward arguments. Let $A \in \mathcal{A}$ denote a generic inference-time search variable, such as stochastic noise perturbations or structured prompt–guidance actions; here A need not denote a single step action A_k. Given $A = a$, the sampler induces a terminal distribution

$$x_0 \sim p_\theta(x_0 \mid a).$$

Let $\rho(a)$ be a reference distribution over actions. The induced marginal output distribution is

$$p_{\theta,\rho}(x_0) = \int_{\mathcal{A}} \rho(a) p_\theta(x_0 \mid a)\, da.$$

The mutual information between the action and generated output is

$$I_{\theta,\rho}(A; X_0) = \mathbb{E}_{A \sim \rho,\ X_0 \sim p_\theta(\cdot \mid A)} \left[\log \frac{p_\theta(X_0 \mid A)}{p_{\theta,\rho}(X_0)} \right].$$

This measures how strongly the searched variable influences the final sample under the reference action distribution.

Let $R = r(X_0)$ be the terminal reward and define

$$\mu(a) := \mathbb{E}[R \mid A = a], \qquad \bar{r}_\rho := \mathbb{E}_{A\sim\rho}[\mu(A)].$$

The quantity $\mu(a)$ is the reward landscape exposed by the searched variable. If $\mu(a)$ is nearly flat, then even an accurate reward model has little exploitable variation to select from.

Theorem E.1 (Mean absolute reward variation)**.** *Assume $R = r(X_0)$ is V-sub-Gaussian under the marginal distribution induced by $p_{\theta,\rho}$. Then*

$$\mathbb{E}_{A\sim\rho}\left[|\mu(A) - \bar{r}_\rho|\right] \le \sqrt{2V\, I_{\theta,\rho}(A;R)} \le \sqrt{2V\, I_{\theta,\rho}(A;X_0)}.$$

Proof. The first inequality follows from a standard variance–information inequality for sub-Gaussian random variables. Intuitively, the conditional mean $\mu(A) = \mathbb{E}[R \mid A]$ cannot vary substantially unless A carries information about R. The second inequality follows from the data-processing inequality because $R = r(X_0)$ is a deterministic function of the generated sample, yielding the Markov chain

$$A \to X_0 \to R.$$

Therefore,

$$I_{\theta,\rho}(A;R) \le I_{\theta,\rho}(A;X_0).$$

Combining the two inequalities gives the result. □

This theorem should be interpreted as a capacity statement rather than a guarantee that higher mutual information always improves reward. Higher mutual information means that the searched variable has greater control over the final output; the realized improvement still depends on whether the induced variation is reward-relevant and whether the search algorithm can exploit it. In our setting, prompt conditioning and guidance strength directly enter the denoising network and modify the conditional generation dynamics, making them natural high-influence variables for reward-guided search.

F Information-Theoretic Formulation of Inference-Time Search

Modeling inference-time search. We use the same fixed reference policy ρ, original-prompt random variable C_0, and complete joint-action trajectory $\mathbf{B}$ as in Theorem A.1. The terminal reward is $R = r(X_0, C_0)$. Conditional mutual information $I_{\theta,\rho}(\mathbf{B}; R \mid C_0)$ measures reward dependence under this reference distribution; it does not measure all controllable image variation. Throughout the proof, $p_{\theta,\rho}$ denotes distributions induced by the frozen generator and this fixed reference policy, rather than an adaptive MCTS policy $\pi_k^{(n)}$.

Reward variation bound

We prove the conditional reward-variation bound stated in Theorem A.1.

Proof. Fix a prompt $C_0 = c$. Since the terminal reward is the deterministic mapping

$$R = r(X_0, c),$$

the clean-image distributions $p_{\theta,\rho}(x_0 \mid c)$ and $p_{\theta,\rho}(x_0 \mid \mathbf{b}, c)$ induce the corresponding reward distributions

$$p_{\theta,\rho}(r \mid c) \qquad \text{and} \qquad p_{\theta,\rho}(r \mid \mathbf{b}, c).$$

Define the marginal expected reward by

$$\bar{r}_\rho(c) := \mathbb{E}_{R\sim p_{\theta,\rho}(\cdot|c)}[R] = \int r(x_0, c)\, p_{\theta,\rho}(x_0 \mid c)\, dx_0,$$

and the expected reward conditional on the joint-action trajectory $\mathbf{b}$ by

$$\mu_\rho(\mathbf{b}, c) := \mathbb{E}_{R\sim p_{\theta,\rho}(\cdot|\mathbf{b},c)}[R] = \int r(x_0, c)\, p_{\theta,\rho}(x_0 \mid \mathbf{b}, c)\, dx_0.$$

Because

$$p_{\theta,\rho}(x_0 \mid c) = \int p_{\theta,\rho}(x_0 \mid \mathbf{b}, c)\, p_{\theta,\rho}(\mathbf{b} \mid c)\, d\mathbf{b},$$

we have

$$\bar{r}_\rho(c) = \mathbb{E}_{\mathbf{B}\sim p_{\theta,\rho}(\cdot|c)}\left[\mu_\rho(\mathbf{B}, c)\right].$$

By the variational representation of the KL divergence, for every $\lambda \in \mathbb{R}$,

$$\begin{aligned} &D_{\mathrm{KL}}\left(p_{\theta,\rho}(r \mid \mathbf{b}, c) \,\|\, p_{\theta,\rho}(r \mid c)\right) \\ &\quad \geq \lambda\left(\mu_\rho(\mathbf{b}, c) - \bar{r}_\rho(c)\right) - \log \mathbb{E}_{R\sim p_{\theta,\rho}(\cdot|c)}\left[e^{\lambda(R-\bar{r}_\rho(c))}\right]. \end{aligned}$$

Because the marginal reward distribution $p_{\theta,\rho}(r \mid c)$ is V-sub-Gaussian,

$$\log \mathbb{E}_{R\sim p_{\theta,\rho}(\cdot|c)}\left[e^{\lambda(R-\bar{r}_\rho(c))}\right] \leq \frac{V\lambda^2}{2}.$$

Therefore,

$$\begin{aligned} &D_{\mathrm{KL}}\left(p_{\theta,\rho}(r \mid \mathbf{b}, c) \,\|\, p_{\theta,\rho}(r \mid c)\right) \\ &\quad \geq \lambda\left(\mu_\rho(\mathbf{b}, c) - \bar{r}_\rho(c)\right) - \frac{V\lambda^2}{2}. \end{aligned}$$

Optimizing the right-hand side over $\lambda \in \mathbb{R}$, with

$$\lambda^\star = \frac{\mu_\rho(\mathbf{b}, c) - \bar{r}_\rho(c)}{V},$$

gives

$$\left(\mu_\rho(\mathbf{b}, c) - \bar{r}_\rho(c)\right)^2 \leq 2V D_{\mathrm{KL}}\left(p_{\theta,\rho}(r \mid \mathbf{b}, c) \,\|\, p_{\theta,\rho}(r \mid c)\right).$$

Taking expectation over the joint-action trajectory $\mathbf{B} \sim p_{\theta,\rho}(\cdot \mid c)$ gives

$$\begin{aligned} &\mathbb{E}_{\mathbf{B}\sim p_{\theta,\rho}(\cdot|c)}\left[\left(\mu_\rho(\mathbf{B}, c) - \bar{r}_\rho(c)\right)^2\right] \\ &\quad \leq 2V\, \mathbb{E}_{\mathbf{B}\sim p_{\theta,\rho}(\cdot|c)}\left[D_{\mathrm{KL}}\left(p_{\theta,\rho}(r \mid \mathbf{B}, c) \,\|\, p_{\theta,\rho}(r \mid c)\right)\right]. \end{aligned}$$

Since

$$\bar{r}_\rho(c) = \mathbb{E}_{\mathbf{B}\sim p_{\theta,\rho}(\cdot|c)}\left[\mu_\rho(\mathbf{B}, c)\right],$$

the left-hand side is

$$\mathrm{Var}_{\mathbf{B}\sim p_{\theta,\rho}(\cdot|c)}\left(\mu_\rho(\mathbf{B}, c)\right).$$

By the definition of conditional mutual information, the right-hand expectation is

$$\mathbb{E}_{\mathbf{B}\sim p_{\theta,\rho}(\cdot|c)}\left[D_{\mathrm{KL}}\left(p_{\theta,\rho}(r \mid \mathbf{B}, c) \,\|\, p_{\theta,\rho}(r \mid c)\right)\right] = I_{\theta,\rho}\left(\mathbf{B}; R \mid C_0 = c\right).$$

Hence, for each fixed prompt c,

$$\mathrm{Var}_{\mathbf{B}\sim p_{\theta,\rho}(\cdot|c)}\left(\mu_\rho(\mathbf{B}, c)\right) \leq 2V I_{\theta,\rho}\left(\mathbf{B}; R \mid C_0 = c\right).$$

Finally, averaging over $C_0 \sim p_C$ yields

$$\mathbb{E}_{C_0\sim p_C}\left[\mathrm{Var}_{\mathbf{B}\sim p_{\theta,\rho}(\cdot|C_0)}\left(\mu_\rho(\mathbf{B}, C_0)\right)\right] \leq 2V I_{\theta,\rho}\left(\mathbf{B}; R \mid C_0\right).$$

□

The bound above concerns conditional reward variation under a fixed reference policy. It does not by itself bound the improvement obtained by changing that policy relative to an arbitrary default sampler; such a claim would require assumptions linking the two distributions.

G NOISE-SPACE SEARCH AND ITS LIMITATIONS

Many gradient-free diffusion search methods define exploration through stochastic noise perturbations. Here t indexes consecutive reverse steps, so $t-1$ denotes the next step; the main formulation uses the explicit schedule t_k. A broad class of stochastic reverse samplers can be written as

$$x_{t-1} = F_\theta(x_t, c_0, \omega_0) + \sigma_t z_t, \qquad z_t \sim \mathcal{N}(0, I),$$

where F_θ is the deterministic denoising component and $\sigma_t z_t$ is the stochastic component. Noise-based search explores different realizations of z_t while keeping the denoising map F_θ fixed.

For example, tree-based or evolutionary search may construct a perturbed noise candidate as

$$z'_t = (1-\alpha) z_t + \alpha \tilde{z}_t, \qquad \tilde{z}_t \sim \mathcal{N}(0, I), \quad \alpha \in [0, 1].$$

This can produce useful sample diversity, and therefore noise search can improve with enough samples. However, this exploration is indirect: the conditional denoising dynamics remain fixed, and reward-relevant changes must be discovered through repeated stochastic sampling.

This limitation becomes more pronounced in modern efficient samplers. In few-step distilled models, the denoising horizon is short, so there are few intermediate opportunities to perturb the trajectory. In deterministic ODE or flow-matching samplers, the stochastic term is absent, corresponding to $\sigma_t = 0$, and the reverse update becomes

$$x_{t-1} = F_\theta(x_t, c_0, \omega_0).$$

Once the initial condition is fixed, the trajectory is determined. Noise-based exploration then either reduces to initial-seed resampling or requires explicitly modifying the sampler to reintroduce stochasticity. ActDiff avoids this dependence on native stochastic branching by searching over conditional inputs that remain meaningful in stochastic, deterministic, original, and few-step samplers.

H STATE-DEPENDENT DIAGNOSTICS AND TRAJECTORY-FEEDBACK PRIOR

ActDiff-MCTS uses terminal rewards for value backup, but terminal rewards are sparse and only available after complete generation. To guide early exploration, we use lightweight diagnostics computed from the local denoising behavior of expanded actions.

At a denoising state $s_k = (x_{t_k}, t_k)$, consider an expanded action

$$a_k = (\widetilde{c}_k, \widetilde{\omega}_k).$$

After applying a_k, we compute two diagnostics.

First, the conditioning-response score measures the strength of the prompt-conditioned direction:

$$d_k(s_k, a_k) = \mathrm{RMS}\left(\epsilon_\theta(x_{t_k}, t_k, \widetilde{c}_k) - \epsilon_\theta(x_{t_k}, t_k, \varnothing)\right).$$

Second, the trajectory-deviation score measures how strongly the action changes the local denoising trajectory:

$$u_k(s_k, a_k) = \mathrm{RMS}\left(\hat{x}_0^{\mathrm{child}}(s_k, a_k) - \hat{x}_0^{\mathrm{base}}(s_k)\right).$$

Here, $\hat{x}_0^{\mathrm{child}}$ is the predicted clean sample after applying the candidate action, while $\hat{x}_0^{\mathrm{base}}$ is the predicted clean sample under the default action at the same state. Both quantities are computed in latent space with RMS normalization over latent dimensions.

These diagnostics are not reward predictors. Instead, they summarize the local effect of an action and are used to construct a state-dependent prior for PUCT. Let $\mathcal{A}_{\mathrm{exp}}(s_k)$ denote actions already expanded from state s_k. For expanded actions, we define a feedback score

$$\phi(s_k, a_k) = \mathbf{1}[a_k \in \mathcal{A}_{\mathrm{exp}}(s_k)] \left[-w_u f_u\left(\frac{u_k(s_k, a_k)}{\bar{u}_k}\right) + w_d f_d\left(\frac{d_k(s_k, a_k)}{\bar{d}_k}\right)\right],$$

where k indexes the current decision segment, $\bar{u}_k$ and $\bar{d}_k$ are robust segment-wise reference scales, and f_u, f_d are stabilizing transforms. The update term discourages overly aggressive trajectory changes, while the conditioning term favors actions that meaningfully activate the prompt-conditioned pathway.

Given a base prior $P_0(a_k \mid s_k)$, the trajectory-feedback prior is

$$P_{\mathrm{tf}}(a_k \mid s_k) = \frac{P_0(a_k \mid s_k)\exp(\phi(s_k, a_k)/\tau)}{\sum_{a' \in \mathcal{A}(s_k)} P_0(a' \mid s_k)\exp(\phi(s_k, a')/\tau)}.$$

For unexpanded actions, $\phi(s_k, a_k) = 0$, so they retain the base prior up to normalization. Thus, the prior uses no diagnostic observations for unexpanded actions; it only reweights actions whose local denoising behavior has already been observed.

The PUCT rule used in ActDiff-MCTS is

$$a_k^* = \arg\max_{a_k \in \mathcal{A}(s_k)} \left[Q(s_k, a_k) + c_{\mathrm{puct}} P_{\mathrm{tf}}(a_k \mid s_k) \frac{\sqrt{N(s_k)}}{1 + N(s_k, a_k)} \right],$$

where $Q(s_k, a_k)$ is the Monte Carlo average of terminal rewards for rollouts passing through edge (s_k, a_k), and $N(s_k)$ and $N(s_k, a_k)$ are node and edge visit counts.

I Mutual-Information Diagnostic

To empirically verify that structured prompt–guidance actions expose stronger reward-relevant variation than noise-only perturbations, we estimate conditional mutual information using a MINE-style diagnostic (Belghazi et al., 2018). For each prompt, we generate a factorial set of samples over different noise seeds, prompt variants, and CFG scales. Each final image is scored by a fixed reward model.

Let C_0 denote the original prompt, S denote the control channel being evaluated, and R denote the reward. We estimate

$$I(R; S \mid C_0)$$

for several control channels, including noise, CFG-only, prompt-only, and joint prompt–CFG actions.

Here $C_{0,i}$ is the original prompt for sample i, B is the minibatch size, and υ_{C_0} permutes controls within each prompt group. We train a neural critic $T_\eta(R, C_0, S)$ using the Donsker–Varadhan objective. Positive tuples are observed reward–control examples $(R_i, C_{0,i}, S_i)$. Negative tuples are constructed by shuffling controls within the same prompt bucket,

$$(R_i, C_{0,i}, S_{\upsilon_{C_0}(i)}),$$

which approximates the conditional product of marginals $p(r \mid c)p(s \mid c)$. The minibatch objective is

$$\widehat{I}_{\mathrm{DV}} = \frac{1}{B}\sum_{i=1}^{B} T_\eta(R_i, C_{0,i}, S_i) - \log\left[\frac{1}{B}\sum_{i=1}^{B} \exp T_\eta(R_i, C_{0,i}, S_{\upsilon_{C_0}(i)})\right].$$

This diagnostic is used only for analysis and is not required by ActDiff during inference.

J Normalization of Trajectory Diagnostics

The scales of u_k and d_k are strongly dependent on the denoising stage: early segments typically induce larger latent changes, while later segments perform finer refinements. We therefore normalize these diagnostics using segment-wise reference scales. For each decision segment k, we maintain running pools of observed trajectory-deviation and conditioning-response magnitudes and set

$$\bar{u}_k = \max\left(\varepsilon_{\mathrm{floor}}, \mathrm{Percentile}_{75}\{u_k : \text{observed at segment } k\}\right), \tag{16}$$

$$\bar{d}_k = \max\left(\varepsilon_{\mathrm{floor}}, \mathrm{Percentile}_{75}\{d_k : \text{observed at segment } k\}\right). \tag{17}$$

Here $\varepsilon_{\mathrm{floor}} > 0$ is a numerical floor. For a given search node at segment k, these quantities are fixed shared reference anchors across all candidate actions when computing the PUCT prior. Thus, the normalized diagnostics compare actions under the same local scale, rather than changing the reference separately for each action. The 75th percentile provides a robust estimate of the large-but-normal scale at each segment: it is less sensitive to outliers than the maximum, while being less conservative than the median. This normalization allows the feedback prior to compare actions within the same denoising stage without mixing the scale differences between early and late segments.

K Prompt Bank Construction

For each original prompt c_0, we construct a prompt bank

$$\mathcal{C}(c_0) = \{c_0, \tilde{c}^{(1)}, \ldots, \tilde{c}^{(M-1)}\}.$$

Prompt variants are generated under a minimal-edit instruction intended to preserve the core semantics of c_0 while allowing small changes to controllable attributes such as lighting, camera phrasing, style, atmosphere, or mood. This instruction does not guarantee semantic fidelity; preserving entities, counts, negations, and relations requires separate validation. We remove invalid, duplicate, or overly divergent rewrites, and always retain the original prompt as a fallback.

During search, a prompt action selects $\widetilde{c}_k \in \mathcal{C}(c_0)$. All reward models and evaluation metrics are computed with respect to the original prompt c_0, not the selected prompt variant. This keeps the evaluation target fixed; it does not guarantee that rewrites preserve the original task.

L Baseline Details

We use the following notation to describe the training-free test-time scaling baselines. In this section, $c = c_0$ is the fixed original prompt and t indexes consecutive reverse steps. Let $x_T \sim \mathcal{N}(0, I)$ be the initial noise, and

$$x_{t-1} \sim p_\theta(x_{t-1} \mid x_t, c)$$

the frozen reverse transition of the base sampler. For an intermediate state x_t, let $\hat{x}_0(x_t, c)$ denote the predicted clean sample obtained from the model, and let $r(x_0, c)$ be the reward assigned to a terminal image. The baselines below differ in how they allocate additional inference-time compute while keeping p_θ fixed.

Best-of-N (BoN) (Stiennon et al., 2020). Best-of-N is the canonical terminal-selection procedure. It samples N independent reverse trajectories from the base model, evaluates the reward only after full denoising, and returns the highest-scoring terminal sample. Formally, for independent initial noises $\{x_T^{(i)}\}_{i=1}^N$,

$$x_0^{(i)} \sim p_\theta(x_0 \mid c; x_T^{(i)}), \qquad x_T^{(i)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I),$$

and the returned sample is

$$x_0^{\text{BoN}} = x_0^{(i^\star)}, \qquad i^\star = \arg\max_{i \in [N]} r(x_0^{(i)}, c).$$

Thus BoN uses the reward as a terminal verifier and introduces diversity only through independent initial noise seeds.

Beam search (Oshima et al., 2025). Beam search maintains a set of partial denoising trajectories and repeatedly expands and prunes them according to an intermediate reward estimate. Let $\mathcal{B}_t = \{x_t^{(b)}\}_{b=1}^B$ be the beam at timestep t. Each beam element is expanded into K candidate continuations,

$$\mathcal{C}_{t-1} = \left\{ x_{t-1}^{(b,k)} \sim q_{t-1}(\cdot \mid x_t^{(b)}, c) : b \in [B],\ k \in [K] \right\},$$

where q_{t-1} is usually the base reverse kernel or a stochastic variant of it. Each candidate is scored through a predicted clean sample or a short lookahead estimate,

$$s_{t-1}^{(b,k)} = r\left(\tilde{x}_0(x_{t-1}^{(b,k)}, c), c\right), \qquad \tilde{x}_0 \in \{\hat{x}_0, \hat{x}_0^{\text{LA}}\},$$

and the next beam keeps the top-B candidates:

$$\mathcal{B}_{t-1} = \text{TopB}_{(b,k)}\ s_{t-1}^{(b,k)}.$$

The procedure therefore performs greedy path selection over partial denoising trajectories, using intermediate estimates of terminal quality.

Feynman–Kac Steering (FK-Steering) (Singhal et al., 2025). FK-Steering formulates reward-guided generation as an interacting particle system over the reverse process. The desired terminal distribution is a reward-tilted version of the base model,

$$p_{\mathrm{tar}}(x_0 \mid c) \propto p_\theta(x_0 \mid c) \exp\left(\frac{r(x_0, c)}{\alpha}\right),$$

where $\alpha > 0$ controls the strength of the tilt. FK-Steering introduces a sequence of reward-derived potentials $\{G_t\}_{t=0}^T$ over noisy states such that their product approximates the terminal exponential tilt:

$$\prod_{t=0}^{T} G_t(x_t, c) \approx \exp\left(\frac{r(x_0, c)}{\alpha}\right).$$

A particle population $\{x_t^{(i)}, w_t^{(i)}\}_{i=1}^P$ is propagated backward in time. With ancestor index a_i selected by resampling, the generic particle update is

$$x_{t-1}^{(i)} \sim q_{t-1}(\cdot \mid x_t^{(a_i)}, c),$$

$$w_{t-1}^{(i)} \propto G_{t-1}(x_{t-1}^{(i)}, c) \frac{p_\theta(x_{t-1}^{(i)} \mid x_t^{(a_i)}, c)}{q_{t-1}(x_{t-1}^{(i)} \mid x_t^{(a_i)}, c)}.$$

When $q_{t-1} = p_\theta$, steering occurs only through weighting and resampling. When the reward is differentiable, the proposal q_{t-1} can additionally be modified using reward-gradient information.

Diffusion Alignment as Sampling (DAS) (Kim et al., 2025b). DAS casts test-time alignment as approximate sampling from the KL-regularized reward-tilted target

$$p_{\mathrm{tar}}(x) = \frac{1}{Z} p_{\mathrm{pre}}(x) \exp\left(\frac{r(x)}{\alpha}\right).$$

Rather than applying the full reward tilt at once, DAS constructs a tempered sequence of intermediate noisy-state targets. With $\hat{r}(x_t) := r(\hat{x}_0(x_t), c)$ and a nondecreasing temperature schedule $0 = \lambda_T \leq \cdots \leq \lambda_0 = 1$, the intermediate target is

$$\pi_t(x_t) \propto p_t(x_t) \exp\left(\frac{\lambda_t}{\alpha} \hat{r}(x_t)\right).$$

Particles are propagated with an SMC proposal and reweighted between adjacent tempered targets. In the gradient-based form, DAS uses a Gaussian proposal whose mean is shifted in the reward-gradient direction:

$$m_{t-1}(x_{t-1} \mid x_t) = \mathcal{N}\left(\mu_\theta(x_t, t) + \sigma_t^2 \frac{\lambda_{t-1}}{\alpha} \nabla_{x_t} \hat{r}(x_t),\ \sigma_t^2 I\right).$$

The corresponding SMC weight update can be written as

$$w_{t-1} = \frac{\tilde{\gamma}_{t-1}(x_{t-1})}{\tilde{\gamma}_t(x_t)} \frac{L_t(x_t \mid x_{t-1})}{m_{t-1}(x_{t-1} \mid x_t)}, \qquad \tilde{\gamma}_t(x_t) = p_t(x_t) \exp\left(\frac{\lambda_t}{\alpha} \hat{r}(x_t)\right),$$

where L_t is the backward kernel used in the SMC construction. The annealing schedule reduces the variance of the reward tilt by gradually introducing the reward along the reverse trajectory.

Search over Paths (SoP) (Ma et al., 2025a). Search over Paths performs local search over intermediate denoising states. Starting from a noisy state x_σ at noise level σ, the method creates local branches by first applying a forward noising perturbation and then partially denoising each branch. For branch noise $\xi^{(m)} \sim \mathcal{N}(0, I)$, a generic forward perturbation is

$$x_{\sigma+\Delta_f}^{(m)} = \mathcal{F}_{\sigma \to \sigma+\Delta_f}(x_\sigma, \xi^{(m)}), \qquad m = 1, \ldots, M,$$

followed by partial reverse denoising,

$$\bar{x}_{\sigma'}^{(m)} = \Phi_{\sigma+\Delta_f \to \sigma'}\left(x_{\sigma+\Delta_f}^{(m)}; c\right), \qquad \sigma' = \sigma + \Delta_f - \Delta_b.$$

The resulting branches are scored through their predicted clean samples,

$$s^{(m)} = r\left(\hat{x}_0(\bar{x}^{(m)}_{\sigma'}, c), c\right),$$

and the highest-scoring branches are retained:

$$\mathcal{P}_{\sigma'} = \mathrm{TopN}_m \; s^{(m)}.$$

Repeating this branch–denoise–select operation yields a local search procedure over the denoising path. In a single-root specialization, the initial seed is fixed and all additional compute is spent on path-level perturbations from that root.

M Implementation Details

M.1 Mutual Information Estimation

We use MINE (Belghazi et al., 2018) to obtain the estimates in Figure 2.

Relation to the main experiments. The mutual-information diagnostic is computed on a single factorial generation set and does not require running any search baselines. This lets us draw on a larger prompt pool (500 prompts, vs. the 200-prompt evaluation subset used for the main results) for a more stable MINE estimate, while the main tables are capped at 200 prompts because each entry requires a full sweep of methods, backbones, and reward models. Both pools are drawn from the same HPSv2 test distribution. For the same factorial-cost reason we restrict the diagnostic to a 5-value CFG grid $\{1.0, 1.25, 1.5, 1.75, 2.0\}$: extending it to the full 7-value search bank would raise the generation count from $500 \times 4 \times 16 \times 5 = 160{,}000$ to $224{,}000$. The diagnostic is intended only to establish a qualitative ordering across control channels (noise vs. CFG vs. prompt vs. joint) on the specified grid, rather than to reproduce the full search configuration. The five-value grid covers the low-to-mid CFG range; the diagnostic does not establish that the ordering is invariant to grid resolution.

Control channels. We train separate estimators for five control channels:

Channel	S	Fixed variables
Noise only	z	variant $= 0$, $\omega = \omega_0$
CFG only	ω	variant $= 0$
Prompt variant only	$\widetilde{c}$	$\omega = \omega_0$
Variant+CFG	$(\widetilde{c}, \omega)$	none
Noise+Variant+CFG	$(z, \widetilde{c}, \omega)$	none

Here z denotes the noise realization, indexed in the implementation by the random seed, $\widetilde{c}$ denotes the prompt variant, and ω denotes the CFG scale.

Conditional MINE estimator. We employ the Mutual Information Neural Estimator (MINE) (Belghazi et al., 2018) to measure $I(R; S \mid C_0)$. For each channel S, we train a neural critic

$$T_\eta(R, C_0, S) \in \mathbb{R}$$

using the Donsker–Varadhan MINE objective. Positive pairs are observed reward–control tuples

$$(R_i, C_{0,i}, S_i).$$

Negative pairs are constructed by shuffling the controls within the same prompt bucket,

$$(R_i, C_{0,i}, S_{\upsilon_{C_0}(i)}),$$

which approximates the conditional product of marginals

$$p(r \mid c)p(s \mid c).$$

The minibatch estimator is

$$\widehat{I}_{\mathrm{DV}} = \frac{1}{B}\sum_{i=1}^{B} T_\eta(R_i, C_{0,i}, S_i) - \log\left[\frac{1}{B}\sum_{i=1}^{B} \exp T_\eta(R_i, C_{0,i}, S_{\upsilon_{C_0}(i)})\right].$$

Thus, the critic is trained to distinguish observed reward–control dependence from prompt-conditioned shuffled controls, yielding an estimate of $I(R; S \mid C_0)$.

Critic architecture and optimization. The critic input is the concatenation of the scalar reward, a learned prompt embedding, and learned embeddings for each discrete control variable:

$$\left[R,\ e_C(C_0),\ e_S(S^{(1)}), \ldots, e_S(S^{(m)})\right].$$

Here $S^{(j)}$ is coordinate j of a control channel, while S_i above indexes a sample. We use embedding dimension 32 for prompt and control embeddings, followed by a three-layer MLP with hidden dimension 256 and SiLU activations. The critic is trained with AdamW using learning rate 10^{-4}, batch size 1024, gradient clipping at 1.0, and 10k optimization steps. We use an 80/20 row-level train-validation split and run 3 random restarts for each control channel, retaining the best validation checkpoint.

M.2 Sampling Configurations

Table 15: Shared sampling, reward, and prompt-rewrite defaults.

Category	Parameter	Default / Notes
Sampling	Steps	4 for FLUX.1-schnell, SiD-SD3.5, SenseFlow-SD3.5; 28 for SD3.5-Large
	Resolution	1024×1024
	Baseline CFG	1.0 by default; SD3.5-Large: 4.5; FLUX.1-schnell: 0.0
	CFG bank	$\{1.0, 1.25, 1.5, 1.75, 2.0, 2.25, 2.5\}$
	Backend-specific CFG	SD3.5-Large: $\{2.5, 3.5, 4.5, 5.5, 7.0, 9.0\}$; FLUX.1-schnell: $\{1.0, 1.25, 1.5, 1.75, 2.0\}$
	Prompt variants	4

FLUX guidance convention. FLUX.1-schnell uses latent adversarial diffusion distillation; its native example sets `guidance_scale=0` (Black Forest Labs, 2024). This API setting is distinct from the two-branch mixing coefficient ω in Eq. 2: $\omega = 1$ returns the conditional prediction, while $\omega > 1$ extrapolates from the unconditional prediction toward the conditional one. In this convention, the reported FLUX bank $\{1, 1.25, 1.5, 1.75, 2\}$ includes the conditional endpoint and four modest extrapolation levels, capped at twice the conditional–unconditional difference. Using this bank requires an explicit two-branch operation; changing the native guidance parameter alone is not that operation.

N Algorithm Configuration and Auxiliary Costs

Tables 16 and 17 state the two backbone-cost conventions and their configurations.

Table 16: **Per-prompt backbone-forward accounting.** All counts are transformer forward-equivalents of the diffusion backbone, not API-level sampler calls. Reward-model forwards are reported separately.

Method	NFE computation	Budget settings	Backbone NFE	Reward NFE
(a) Few-step distilled models (T=4, ω_0=1.0 so c_0=1, $\Omega = \{1.0, 1.25, 1.5, 1.75, 2.0, 2.25, 2.5\}$, c_{unif}=13/7 ≈ 1.857)				
Teacher baseline (28-step)	$2\,T_{\text{base}}$	T_{base}=28	56	1
Distilled Baseline	$T\,c_0$	ω_0=1.0	4	1
BoN	$N\,T\,c_0$	N=76	304	76
DAS / FK-Steering	$2\,K\,T$	K=8	64	32[§]
DTS / DTS*	$K\,T\,c_{\text{unif}}$	K=8	≈60	32
SoP	$K\,T\,c_{\text{unif}}$	K=8	≈60	32
Beam	$K\,T\sum_{\omega\in\Omega} c(\omega) = 13KT$	K=4, $\lvert\Omega\rvert$=7	208	112
ActDiff (ours)	$\sum_{m=1}^{M}(T - d_m)\,\bar{c}_m$	M=30	≈223	30
(b) Multi-step model (SD3.5-Large, T=28, ω_0=$\langle\omega_0\rangle$, $\Omega = \{3.5, 4.0, 4.5, 5.0, 5.5, 6.0, 7.0\}$; $c(\omega)$=2 for all ω, hence exact counts)				
Baseline	$2\,T$	ω_0=$\langle\omega_0\rangle$	56	1
BoN	$2\,N\,T$	N=136	7616	136
DAS / FK-Steering	$2\,K\,T$	K=8	448	224[§]
DTS / DTS*	$2\,K\,T$	K=8	448	224
SoP	$2\,K\,T$	K=8	448	224
Beam	$2\,K\,\lvert\Omega\rvert\,T$	K=4, $\lvert\Omega\rvert$=7	1568	784
ActDiff (ours)	$2\sum_{m=1}^{M}(T - d_m)$	M=30	996–1005	30

Note: T: denoising steps; $c(\omega) \in \{1, 2\}$: backbone forwards per step under two-branch CFG (c=1 at ω=1.0, c=2 at ω>1.0); c_0=$c(\omega_0)$ at the default guidance ω_0; c_{unif}=13/7: expectation under uniform draws from Ω; $\bar{c}_m$: empirical multiplier measured in simulation m; K: particles or beam width; N: noise samples; M: MCTS simulations; d_m: expansion depth of simulation m. Beam expands every action in the bank at each step, so its count is exact; ≈ marks expectation-based estimates, evaluating the formulas at full simulation depth (d_m=0) under uniform draws from Ω; prefix caching and adaptive guidance selection reduce realized counts below these estimates. Decimal NFEs arise from averaging integer per-prompt counts: selecting CFG = 1 incurs one conditional evaluation per step, whereas CFG > 1 incurs conditional and unconditional evaluations. Different selected scales can therefore yield a fractional dataset mean. Likewise, 13/7 is an expected forward cost, not a fractional network invocation. These forward-equivalent counts differ from the batched-invocation convention in Table 17. Reward NFE counts one reward-model forward per scored image. Beam scores candidates at each step; the listed ActDiff configuration scores terminal images only. If four key-step evaluations per simulation are also used, they add to the M terminal scores ($30 + 30\times4 = 150$). [§] Gradient-guided variants additionally backpropagate through the reward model (one backward pass per scored sample), increasing effective reward cost beyond the listed forward count; DAS and FK-Steering otherwise have identical counts by construction.

Costs outside backbone NFEs. The four-member prompt bank contains the original prompt and three Qwen3-4B rewrites, prepared before trajectory search. Its cost is charged once per bank construction; reusing a bank amortizes that cost across searches. The available experiment records do not report rewriter token counts or wall-clock time. Table 16 reports reward-model forward counts for its stated configurations. A composite evaluation requires one forward per constituent scorer: Q scored images cost $2Q$ forwards for IR+HPSv3 and $3Q$ for IR+HPSv3+PickScore. For the table's 30-simulation, terminal-only ActDiff configuration, these counts are 60 and 90, respectively. Reward-gradient variants also require backward passes. These counts exclude final offline evaluation on held-out scorers. End-to-end latency additionally includes prompt encoding, decoding, and search overhead. Appendix N.1 reports per-prompt wall-clock and memory measurements for a separate four-prompt SiD study, including online reward calls but excluding offline rewrite generation and pipeline loading. Component timings and main-benchmark end-to-end latency remain unreported. Backbone NFEs and reward-call counts quantify separate resources and should be read alongside the measured timing scope.

Table 17: Configured generator cost across few-step and multi-step diffusion models. Exact costs follow deterministically from the fixed population, trajectory, or action-bank configuration. For tree-based methods that reuse cached prefixes, we report an estimated mean NFE followed by the worst-case upper bound in parentheses.

Method	**Configuration**	**Few-step models** ($T = 4$)	**SD3.5-Large** ($T = 28$)	**Cost type**
Distilled / teacher baseline	One complete trajectory	4	28	Exact
FK-Steering	$K = 32$ particles	128	224	Exact
Best-of-N (BoN)	$N = 32$ complete trajectories	128	448	Exact
DAS	$N = 32$ continuous-action trajectories	128	448	Exact
Beam search	$K = 4$ with the prompt–CFG action bank	312 (FLUX) 364 (SiD/SenseFlow)	3,052	Exact
Search over Paths (SoP)	$N = 8$, $M = 8$, $K = 4$; branching begins at 25% progress	148	$\approx 408^{\dagger}$	Configured mean
Genetic algorithm (GA)	Population $P = 24$, generations $G = 8$, plus one final replay	772	5,404	Exact
DTS	$M = 64$ cached tree iterations	$\approx 160^{\dagger}$ (≤ 256)	$\approx 928^{\dagger}$ ($\leq 1{,}792$)	Mean estimate
DTS*	$M = 64$ cached tree iterations	$\approx 160^{\dagger}$ (≤ 256)	$\approx 928^{\dagger}$ ($\leq 1{,}792$)	Mean estimate
Dynamic CFG	One adaptive trajectory	4	28	Exact
ActDiff	25 split simulations, and 2 exploit replays	$\approx 103^{\dagger}$ (≤ 140)	$\approx 643^{\dagger}$ (≤ 980)	Mean estimate

One NFE denotes one batched denoiser invocation for one propagated trajectory step. Conditional and unconditional predictions evaluated together for classifier-free guidance count as one batched invocation. Reward-model evaluations, prompt rewriting, reward backpropagation, and other auxiliary computation are accounted for separately.

† For cached tree and split-search procedures, the estimate assumes that the expansion or split depth is approximately uniform over the denoising horizon, giving an expected uncached suffix length of $\mathbb{E}[T - d] = (T + 1)/2$. Parenthesized values are worst-case bounds obtained when every simulation evaluates a complete trajectory.

N.1 Measured Wall-Clock and GPU Memory

Study scope and setup. We profile four-step SiD-SD3.5-Large at 1024×1024 on one NVIDIA RTX 6000 Ada, with ImageReward-v1.0 as the search reward. The four prompts are the SHA256-ordered first four of a separate 25-prompt cohort (`q0117`, `q0174`, `q0042`, `q0184`); the base seed is 42. Each method uses its own profiling configuration, rather than a matched NFE or latency budget. This study uses cached Qwen2.5-3B rewrites and heuristic stage rewrites; it is distinct from the 200-prompt benchmark with Qwen3-4B. The ActDiff row is the single-seed configuration; timings for its 16-seed multi-root variants are not reported here.

Measurement boundaries. Each method runs in a fresh process. Per-prompt runner times exclude the approximately 60 s pipeline load, offline rewrite generation, and post-hoc judging; image saving is enabled. Prompt encoding, sampling, and online reward calls occur within the timed run. The separate ImageReward server shares the GPU. Eleven of the cheaper methods were repeated in a second pass: the recorded times agree to a median of 2.4% and a maximum of 12.7%; the reported per-prompt coefficient of variation is 0.2–9.4%. These four-prompt measurements characterize the listed implementation and configurations. They do not provide timing uncertainty for the main benchmark or separate the time spent in rewriting, decoding, and reward scoring.

Table 18: **Wall-clock and search-phase memory on SiD-SD3.5.** Methods use their own profiling defaults. Seconds are mean ± s.d. over four prompts; NFE counts transformer forward passes in these runs. The Dynamic CFG NFE was not reported. Peaks use the sampler's allocated-memory counter during search (or the trajectory for the two no-search rows), excluding the reward server. Table 19 reports encoding and process-level memory.

Method	NFE	s / prompt	× baseline	Peak GiB
Baseline (no search)	4	10.2 ± 1.0	1.0	17.7
Dynamic CFG (x_0 scoring)	–	15.1 ± 1.6	1.5	17.7
Best-of-N	64	46.5 ± 1.6	4.6	17.7
DAS (continuous actions)	128	67.8 ± 1.4	6.7	17.7
Greedy over rewrites	12	26.2 ± 1.3	2.6	17.7
Greedy (step-wise argmax)	208	165.6 ± 3.8	16.3	17.8
Beam search	676	506.6 ± 7.4	49.9	17.8
SMC / DAS-tempering	64	52.4 ± 1.8	5.2	17.7
FK-Steering	64	52.8 ± 1.5	5.2	17.7
SoP	72	87.8 ± 3.0	8.7	17.8
SoP + CFG + rewrite	120	105.5 ± 2.4	10.4	17.8
ActDiff (improved)	52	51.5 ± 3.6	5.1	17.8
DTS*	105	76.7 ± 2.7	7.6	17.7
DTS	110	82.7 ± 6.5	8.1	17.7
ActDiff (unified root)	164	88.5 ± 2.4	8.7	17.8
ActDiff (UCB1)	185	95.6 ± 2.2	9.4	17.8
ActDiff + adaptive CFG + rewrites	194	101.7 ± 5.0	10.0	17.8
Genetic algorithm	956	477.4 ± 37.0	47.0	17.7

Observed cost. In Table 18, per-prompt time ranges from 10.2 s for one baseline trajectory to 506.6 s for beam search. BoN uses 64 NFEs and 46.5 s, while the single-seed ActDiff + adaptive CFG + rewrites configuration uses 194 NFEs and 101.7 s. NFE alone does not determine latency: for example, SoP uses 72 NFEs and 87.8 s, versus 128 NFEs and 67.8 s for continuous-action DAS. Decoding, reward calls, and search bookkeeping contribute additional work, so comparisons should retain both the method configuration and its measured time.

Memory measurement and scope. We reset `torch.cuda.max_memory_allocated` at encoding, baseline, and search boundaries. Search is sequential; its largest recorded memory increase over a baseline trajectory is 61 MiB (beam). A separate 1 Hz `nvidia-smi` poller measures the sampler's process footprint, including allocator reservation. All sampler figures exclude the separate 2.2 GiB ImageReward server on the same GPU. Batching candidates changes these memory requirements.

Table 19: **Memory by phase in the SiD profiling study.** Allocated peaks are maxima over four prompts, with ranges across methods; the search row covers phase-traced methods. The process footprint includes allocator reservation. The separate 2.2 GiB reward server is excluded.

Phase or measurement	GiB
Resident after prompt encoding (transformer + VAE, fp16)	15.3
Prompt encoding (T5-XXL + CLIP on GPU)	30.3
Baseline trajectory (4 steps + VAE decode + reward call)	17.7–17.8
Search phase	17.7–17.8
Sampler process footprint from `nvidia-smi`	30.9

O Full Multi-Reward Benchmark Comparisons

Tables 20 and 21 give the full method comparisons for IR + HPSv3 and IR + HPSv3 + PickScore, respectively, with each backbone grouped across columns. These preference composites are distinct from the set of preference and alignment scores examined in the orthogonality study in Section 4.3.

Table 20: ImageReward + HPSv3 search across FLUX.1-schnell (Black Forest Labs, 2024), SiD-SD3.5 (Zhou et al., 2025), and SenseFlow-SD3.5 (Ge et al., 2026). Higher reward scores are better.

	FLUX.1-schnell					SiD-SD3.5					SenseFlow-SD3.5				
Method	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick
Baselines															
Multi-step Baseline	56	1.269	0.308	13.613	22.784	56	1.022	0.308	12.240	22.807	56	1.022	0.308	12.240	22.807
Distilled Baseline	**4**	1.213	0.308	13.558	22.801	**4**	1.180	0.308	12.340	22.756	**4**	1.269	0.308	12.220	22.652
Multiple-root search															
DAS*	256	1.447	0.313	14.592	22.812	256	1.399	0.310	13.435	22.821	256	1.424	0.311	13.475	22.807
FK-Steering*	256	1.447	0.313	14.592	22.820	256	1.400	0.311	13.470	22.816	256	1.424	**0.314**	13.546	22.817
BoN	256	1.404	0.312	14.636	22.809	256	1.422	0.309	13.786	22.832	256	1.412	0.310	13.898	22.830
Beam	455	1.341	0.306	14.091	22.783	455	1.144	0.307	12.215	22.794	455	1.254	0.298	12.186	22.815
Single-root search															
SoP	60	1.403	**0.314**	14.824	22.831	60	1.407	**0.312**	13.981	22.821	60	1.442	0.310	14.137	22.821
GA	768	1.256	0.305	14.218	22.825	768	1.102	0.306	12.730	22.761	768	1.297	0.310	12.910	22.785
DTS	256	1.229	0.303	13.095	22.762	256	1.220	0.305	12.386	22.754	256	1.261	0.310	12.360	22.763
DTS*	256	1.413	0.306	14.685	22.807	256	1.427	**0.312**	13.762	22.814	256	1.412	0.310	13.890	22.793
Dynamic CFG†	8	1.241	0.304	13.803	22.802	8	1.164	0.310	12.380	22.746	8	1.290	0.310	12.470	22.766
ActDiff	240	**1.467**	**0.314**	**15.024**	**22.853**	246.7	**1.455**	0.310	**14.109**	**22.845**	248.4	**1.450**	0.310	**14.408**	**22.843**

Pick denotes PickScore. The objective weights IR and HPSv3 equally. HPSv2 and PickScore are held out. Bold reward scores mark column maxima among search methods; ties are bold. *DAS/FK-Steering use reward gradients where feasible and gradient-free variants otherwise. †Dynamic CFG searches guidance only.

Table 21: ImageReward + HPSv3 + PickScore search across FLUX.1-schnell (Black Forest Labs, 2024), SiD-SD3.5 (Zhou et al., 2025), and SenseFlow-SD3.5 (Ge et al., 2026). Higher reward scores are better.

	FLUX.1-schnell					SiD-SD3.5					SenseFlow-SD3.5				
Method	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick	NFE	IR	HPSv2	HPSv3	Pick
Baselines															
Multi-step Baseline	56	1.269	0.308	13.613	22.784	56	1.022	0.308	12.240	22.807	56	1.022	0.308	12.240	22.807
Distilled Baseline	**4**	1.213	0.308	13.558	22.801	**4**	1.180	0.308	12.340	22.756	**4**	1.269	0.308	12.220	22.652
Multiple-root search															
DAS*	256	1.532	0.310	13.537	22.863	256	1.518	0.309	12.680	22.798	256	1.536	0.310	12.566	22.810
FK-Steering*	256	1.540	0.311	13.526	22.844	256	1.537	0.309	12.674	22.818	256	1.567	0.310	12.593	22.813
BoN	256	1.565	0.313	13.414	22.845	256	1.569	0.312	12.828	22.901	256	1.562	0.311	12.865	22.981
Beam	455	1.412	0.307	13.548	22.833	455	1.359	0.306	10.862	22.312	455	1.438	0.307	12.328	22.879
Single-root search															
SoP	60	1.590	**0.314**	13.536	22.781	60	1.623	0.310	12.198	22.635	60	1.605	0.312	12.827	22.934
DTS	256	1.278	0.307	13.227	22.739	256	1.272	0.309	12.225	22.781	256	1.294	0.306	12.249	22.838
DTS*	256	1.581	0.313	13.648	22.833	256	1.598	**0.314**	12.774	22.880	256	1.575	0.312	12.735	22.955
ActDiff	240	**1.629**	0.313	**13.856**	**22.909**	246.7	**1.625**	0.311	**12.851**	**22.910**	248.4	**1.644**	**0.313**	**12.976**	**22.998**

Pick denotes PickScore. The objective weights IR, HPSv3, and PickScore equally. HPSv2 is held out. GA and Dynamic CFG were not run. Bold reward scores mark column maxima among search methods; ties are bold. *DAS/FK-Steering use reward gradients where feasible and gradient-free variants otherwise.

P Reward Orthogonality: Protocol and Interpretation

P.1 Evaluation cohort

The study uses 200 prompts: 100 from GenAI-Bench (Li et al., 2024a), spanning counting, comparison, negation, and spatial relations, and 100 from the four aesthetic styles in the HPSv2 benchmark (Wu et al., 2023). Four native samples per prompt are generated on each of FLUX.1-schnell, SD3.5-Large, SiD-SD3.5-Large, and SenseFlow-SD3.5-Large, giving $200 \times 4 \times 4 = 3{,}200$ images. This cohort is separate from the 200-HPSv2-prompt search benchmark. Native sampling differs across backbones, so comparisons between backbones also reflect sampling-regime differences.

P.2 Preference and alignment scores

HPSv3 and ImageReward provide learned preference scores (Ma et al., 2025b; Xu et al., 2023). VQAScore uses CLIP-FlanT5-XXL to score the probability of answering "Yes" to a question about whether the image depicts the original prompt (Lin et al., 2024). It is not a hard binary correctness score or an average of separately generated condition-level questions. All three scorers use the original prompt.

P.3 Agreement and effective dimensionality

The reported correlations characterize variation within prompt groups, rather than pooling differences in prompt difficulty. HPSv3–ImageReward Pearson correlation is 0.32 (95% CI $[0.25, 0.38]$), ImageReward–VQAScore is 0.28 ($[0.18, 0.37]$), and HPSv3–VQAScore is 0.11 ($[0.05, 0.17]$). The last correlation is -0.001 on SiD and falls to 0.025 overall when controlling for ImageReward. The participation ratio is 2.66 out of three possible dimensions (2.84 on SiD).

P.4 Scope of the evidence

These statistics measure complementary variation within the evaluated candidate sets. They characterize score agreement, rather than statistical independence or accuracy relative to human preferences. Disagreement can reflect both distinct evaluation criteria and evaluator error. The reported confidence intervals are treated as nominal; the available summary does not specify all details needed to reproduce their resampling procedure.

This observational analysis does not measure robustness to increasing search pressure or the effectiveness of a mitigation intervention. The main multi-reward benchmark uses preference composites containing IR, HPSv3, and optionally PickScore. Its results and this VQAScore-based complementarity study should be interpreted separately.