DUALHEADS: REWARD-TILTED FEW-STEP DIFFUSION MADE SIMPLE

Anonymous authors
Paper under double-blind review



Figure 1: **Four-step samples from DualHeads.** Z-Image-Turbo (Jiang et al., 2026) fine-tuned with DualHeads using the HPSv3 reward (Ma et al., 2025).

ABSTRACT

Distilled diffusion models generate high-quality images in a few sampling steps, but aligning them with task requirements and human preferences is difficult. Methods developed for multi-step diffusion models rely on differentiable rewards, or on score fields and stochastic trajectories that distilled generators may lack. We ask whether a few-step generator can be adapted using only external reward evaluations and reference samples, and show that two binary classifiers suffice. Our method, DualHeads, implements them as lightweight heads on a frozen diffusion backbone. The reward head learns which samples of the same prompt score higher, the reference head separates reference images from generated ones, and the generator maximizes a weighted sum of their logits. When the heads are optimal, this objective is KL-regularized maximization of the learned reward, and the ratio of head weights sets the regularization strength. Empirically, with both rule-based and model-based rewards, a few-step DualHeads generator reaches task rewards close to those of 40-step reinforcement learning methods while preserving image quality, at no extra inference cost. We hope that this simple and elegant dual-head formulation will help advance few-step reward-tilted generation.

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.

1 INTRODUCTION

Diffusion distillation (Song et al., 2023; Yin et al., 2024b; Sauer et al., 2024a; Zhou et al., 2024) turns powerful diffusion models (Song et al., 2020; Ho et al., 2020; Rombach et al., 2022; Liu et al., 2022; Lipman et al., 2022) into few-step generators that produce high-quality samples in a handful of network evaluations, bringing applications such as real-time interactive video generation within reach (Yin et al., 2025; Huang et al., 2026a).

Speed alone, however, does not make a sample correct: a distilled student inherits the teacher's failures, and may still misspell the prompted text, miscount objects, or ignore the requested style. Such failures are readily caught by external evaluators, such as text recognizers, object detectors, or preference models. Using these external evaluators as rewards to post-train a generator is now standard practice for diffusion models (Lee et al., 2023; Xu et al., 2023; Prabhudesai et al., 2024).

Existing methods fall into two families, each one addressing typically only one side of this setting. First, reinforcement learning (RL) with black-box rewards has succeeded for multi-step diffusion models, through policy gradients that reinforce high-reward denoising steps (Black et al., 2024; Fan et al., 2023; Liu et al., 2025; Xue et al., 2025b) or forward-process objectives (Zheng et al., 2025; Xue et al., 2025a; Choi et al., 2026a; Lee et al., 2026). These, however, rely on what distillation removes: policy gradients estimate updates from stochastic multistep rollouts, which become high-variance with only a few steps, and forward-process objectives need a score or velocity field that a distilled generator need not preserve. Second, methods designed for few-step generators pay for the step constraint elsewhere: they backpropagate through a differentiable reward (Luo, 2024) which may not always be feasible, match the generator to the original multi-step model (the teacher) using its score and a second diffusion model fitted to the generator's samples (e.g., Luo et al., 2026; Jiang et al., 2026; Huang et al., 2026b), or adapt multi-step RL specifically to flow-map generators such as consistency models and MeanFlow (Li et al., 2026c; Huang et al., 2026c). Methods from both families rely on something beyond samples and black-box scores: trajectory likelihoods, a score field, reward gradients, or the teacher's score.

In practice, however, the only thing any few-step generator is guaranteed to provide is samples. This leads to our central question: *given only its own samples, black-box reward evaluations, and examples of high-quality images from a reference model, can we tilt a few-step generator toward higher reward regions while keeping it close to those images?*

We show that this can be done with only one backbone network and two binary classifiers. We begin with the standard reference-regularized RL objective (Levine, 2018), which maximizes the expected reward while penalizing the reverse KL divergence to the reference distribution to prevent reward hacking. Its optimum is the reference distribution reweighted by the exponentiated reward. The two terms are, however, hard to optimize directly in our setting, since we treat the reward as a black box and the KL term depends on the density ratio between the generator and the reference. Our key observation is that both can be replaced by quantities learned through classification.

This leads to DualHeads, a simple framework consisting of two identical lightweight heads on a shared frozen diffusion backbone. A *reward head* learns which samples are better from their black-box rewards: for multiple generations of the same prompt, rewards are normalized within the group and converted into soft optimality labels. A *reference head* distinguishes noised reference images from noised generator predictions. Both heads are trained with the same binary cross-entropy objective and updated online as the generator updates. To update the generator, we evaluate the two heads on a noised generator prediction and maximize a weighted sum of their logits. Gradients pass through both heads and a single generator forward pass; the sampled trajectory and the head parameters remain fixed. The heads are discarded at inference. We note that at the optimum of the reference head, maximizing this weighted sum is equivalent to the reference-regularized objective from above, with the reward-head logit in place of the reward and the ratio of head weights setting the regularization strength.

We evaluate DualHeads at a four-step setup on complementary modern reward functions, including visual text rendering (OCR), compositional generation (GenEval), and human preference (HPS, PickScore, etc.). Compared with existing methods, DualHeads achieves on-par performance in the few-step regime while preserving generation quality. It raises OCR accuracy from 0.59 to 0.92 and GenEval from 0.63 to 0.90 over the base model, and gives the best HPSv3 on Z-Image.

Our contributions are:

- We introduce DualHeads, a simple framework that adapts few-step generators from reward evaluations and reference samples alone, using two classifier heads on a frozen backbone.
- We show that, with optimal heads, the weighted-logit generator objective is a reverse-KL-regularized reward objective whose temperature is set by the head-weight ratio.
- We provide empirical results with both rule-based and model-based rewards, showing that a four-step student reaches task rewards close to 40-step RL methods while preserving quality.

2 Related Work

Step distillation of diffusion models. Progressive and consistency distillation (Salimans & Ho, 2022; Song et al., 2023) turn a pretrained diffusion model into a few-step generator that covers the sampling trajectory in a small number of large jumps. Latent consistency models (Luo et al., 2023) extend this to text-to-image latent diffusion. Distribution matching distillation (DMD) (Yin et al., 2024b) instead matches the student's output distribution to the teacher's by following the gradient from a score difference, which requires an auxiliary network that tracks the student's score online. DMD2 (Yin et al., 2024a) stabilizes this estimator and adds an adversarial term on real data, and TDM (Luo et al., 2025a) extends the matching to every step of the sampling trajectory. Parallel work relies on adversarial supervision. ADD (Sauer et al., 2024b) distills with a discriminator on pretrained image features, and LADD (Sauer et al., 2024a) moves the discriminator onto frozen features of the teacher diffusion model itself. DualHeads is built on top of a few-step distilled diffusion model. Our reference head follows LADD's design of heads on frozen diffusion features.

Reward optimization of diffusion models. Methods for optimizing a reward differ mainly in how the reward signal reaches the generator. Differentiable rewards can be backpropagated through directly (Xu et al., 2023; Clark et al., 2024; Prabhudesai et al., 2023), but black-box rewards cannot. Reinforcement learning (RL) removes this requirement: DDPO (Black et al., 2024) and DPOK (Fan et al., 2023) use policy gradients, and Flow-GRPO (Liu et al., 2025) and DanceGRPO (Xue et al., 2025b) adapt group-relative policy optimization (Shao et al., 2024) to flow models. These methods need transition likelihoods along stochastic multistep rollouts, which recent work avoids by training on the forward process. DiffusionNFT (Zheng et al., 2025) contrasts positive and negative samples under clipped group-relative optimality labels, CRD (Zhu et al., 2026) matches prompt-centered rewards to cancel an intractable normalizer, and AWM (Xue et al., 2025a), RAM (Bergmeister et al., 2026), RVM (Choi et al., 2026a), and DGPO (Luo et al., 2025b) use related advantage- or reward-weighted targets. In all of them, the reward serves only as a sample weight or a regression target, and no explicit reward model is learned. DualHeads uses labels of the same form as DiffusionNFT, but fits them with an explicit classifier and guides the student through that classifier's input gradient.

Reward optimization coupled with distillation. Other methods optimize the reward inside a distillation pipeline and differ mainly in what keeps the generator close to good images while the reward is optimized. Most rely on a teacher. Some combine a reward objective with distribution matching (Jiang et al., 2026; Chen et al., 2026; Dong et al., 2026; Wang et al., 2026; Fan et al., 2026); for example, RTDMD (Huang et al., 2026b) minimizes a KL divergence to a reward-tilted teacher. Closest to our setting, TDM-R1 (Luo et al., 2026) post-trains a few-step student with non-differentiable rewards by learning a diffusion-model surrogate reward, and regularizes it with trajectory distribution matching, which requires the teacher and an online fake score. Flow-OPD (Fang et al., 2026) and DiffusionOPD (Li et al., 2026a) first train task-specific teachers with RL and then distill them into one student, without reducing the step count. REST (Li et al., 2026b) co-trains an eight-step student with an evolving teacher. Others do without a teacher. MeanFlowNFT (Huang et al., 2026c) applies forward-process RL to a distilled four-step generator, anchored to a frozen EMA of itself, and Self-OPD (Zhang et al., 2026) scores branched student rollouts against the student's own deterministic trajectory. Reward Lightning (Cheng et al., 2026) is closest in architecture: it jointly distills a video teacher into a few-step generator with discriminator and reward heads on one frozen backbone, but its reward head is pretrained on preference pairs and kept frozen. DualHeads instead post-trains a distilled generator and refits its reward head online, on pointwise group-relative labels from black-box rewards, while its reference head keeps the generator close to the reference pool.

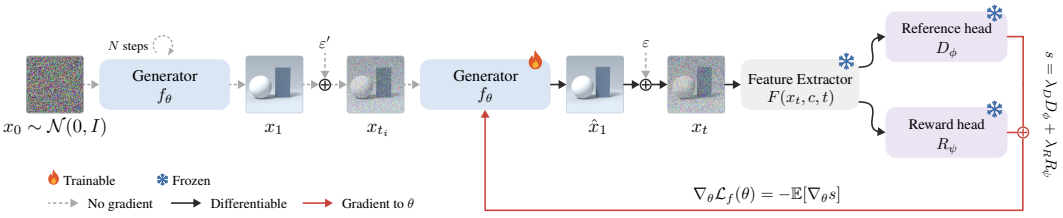


Figure 2: **Generator update of DualHeads.** The generator f_θ first rolls out N steps from noise without gradient; the endpoint x_1 is re-noised to x_{t_i}, from which f_θ predicts $\hat{x}_1$ in one differentiable step. The prediction is noised to x_t and sent to a frozen feature extractor F followed by a reference head D_ϕ and a reward head R_ψ; the generator ascends the utility $s = \lambda_D D_\phi + \lambda_R R_\psi$ through $\hat{x}_1$, while the backbone and both heads stay fixed during its update.

3 METHOD

DualHeads adapts a few-step generator so that its samples score higher under an external reward while staying close to a reference distribution. For this, we use two binary classifiers on a shared frozen backbone: one learns *reference fidelity*, the other *reward optimality*. Section 3.1 describes the few-step generation setting and the corresponding reward-tilted target. Section 3.2 defines the two classifier objectives and Section 3.3 gives the single utility the generator maximizes.

3.1 BACKGROUND

Setup of few-step generator. Let θ denote the parameters of a neural network few-step generator f_θ; c the frozen prompt embeddings and x an image sample. We write time in the flow-matching convention, with data at $t = 1$. For a clean sample x_1 and noise $\epsilon \sim \mathcal{N}(0, I)$, the interpolant at time $t \in [0, 1]$ and the generator's clean-data prediction are:

$$x_t \;=\; \mathcal{N}_t(x_1; \epsilon) = t\, x_1 + (1 - t)\, \epsilon, \qquad f_\theta(x_t, c, t) \;=\; x_t + (1 - t)\, v_\theta(x_t, c, t), \tag{1}$$

where v_θ is the velocity representation of the model output. Starting from Gaussian noise $x_{t_0} \sim \mathcal{N}(0, I)$, the N-step stochastic sampler follows a timestep grid $0 = t_0 < t_1 < \cdots < t_N = 1$. At each step, the model predicts the clean endpoint and re-noises it to the next timestep:

$$x_{t_{i+1}} = \mathcal{N}_{t_{i+1}}(f_\theta(x_{t_i}, c, t_i); \epsilon_{i+1}), \qquad i = 0, \ldots, N - 1, \tag{2}$$

where $\epsilon_1, \ldots, \epsilon_N \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I)$ are independent Gaussian noise. The noising operator satisfies $\mathcal{N}_1(x; \epsilon) = x$ so the final step adds no noise and returns $x_{t_N} = f_\theta(x_{t_{N-1}}, c, t_{N-1})$. We write $p_\theta(x|c)$ for the distribution of the sampler's output $x = x_{t_N}$.

Goal: a reward-tilted reference distribution. We are given a black-box image reward $r(x, c) \in [0, 1]$ and prompt-paired reference samples from a conditional distribution $p_{\text{ref}}(x|c)$. Our goal is to train a few-step generator whose output distribution $p_\theta(x|c)$ achieves high expected reward while remaining close to the reference distribution. For a fixed prompt c, a standard formulation for KL regularized RL training is:

$$\mathcal{J}_c(\theta) = \mathrm{KL}(p_\theta(\cdot|c) \,\|\, p_{\text{ref}}(\cdot|c)) - \frac{1}{\beta} \mathbb{E}_{x \sim p_\theta(\cdot|c)} \big[r(x, c) \big], \tag{3}$$

$$p^*(x|c) = \frac{1}{Z(c)} p_{\text{ref}}(x|c) \exp\left(\frac{r(x, c)}{\beta} \right), \tag{4}$$

where $\beta > 0$ controls the strength of reference regularization relative to reward, $p^*(x|c)$ is the optimal solution to Eq. (3), and $Z(c)$ is the normalizing constant. Indeed, $\mathcal{J}_c(\theta) = \mathrm{KL}(p_\theta \| p^*) - \log Z(c)$, so minimizing $\mathcal{J}_c$ is equivalent to fitting the reward-tilted target in reverse KL.

3.2 DUALHEADS

Since the KL term requires unavailable density ratios and the black-box reward can be non-differentiable, neither term in Eq. (3) is directly tractable. To address this, we propose to learn a classifier surrogate for each: a reference head and a reward head on a shared frozen backbone.

Two heads on one frozen backbone. A pretrained frozen network serves as a feature extractor $F(x_t, c, t)$ for two lightweight heads with identical architecture: The *reference head* D_ϕ distinguishes reference images $x_{\text{ref}} \sim p_{\text{ref}}$ and predictions coming from online generator predictions. The *reward head* R_ψ instead predicts how good a sample is based on some external evaluation score. Its target is a soft label $y \in [0, 1]$, computed from the reward of the rollout endpoint. Both heads are trained with the same binary cross-entropy loss, $\text{BCE}(x; y) = -y \log \sigma(x) - (1-y) \log\big(1-\sigma(x)\big)$. For brevity, we use $D_\phi(x_t, c, t)$ for $D_\phi(F(x_t, c, t), c, t)$ and likewise for R_ψ.

The generator prediction $\hat{x}_1$ is formed in two steps: a rollout endpoint x_1, sampled using Eq. (2), is re-noised to a rollout grid timestep t_i, yielding $x_{t_i} = \mathcal{N}_{t_i}(x_1; \epsilon')$, and the generator is then applied in one differentiable step, yielding $\hat{x}_1 = f_\theta(x_{t_i}, c, t_i)$.

The reference head uses hard targets: 1 for noised reference images, 0 for noised generator predictions, and its training objective is:

$$\mathcal{L}_D(\phi) = \mathbb{E}_{t,\epsilon,x_{\text{ref}}}[\text{BCE}(D_\phi(\mathcal{N}_t(x_{\text{ref}}; \epsilon), c, t); 1)] + \mathbb{E}_{t,\epsilon,\hat{x}_1}[\text{BCE}(D_\phi(\mathcal{N}_t(\hat{x}_1; \epsilon), c, t); 0)]\,. \quad (5)$$

The reward head uses soft label targets y, which express reward preference *relative to the other samples drawn for the same prompt*; we take y as given here and show our construction of it at the end of this section. The loss is identical in form to Eq. (5)[1]:

$$\mathcal{L}_R(\psi) = \mathbb{E}_{t,\epsilon,(x_1,y)}[\text{BCE}(R_\psi(\mathcal{N}_t(x_1; \epsilon), c, t); y)]\,. \quad (6)$$

The label admits a probabilistic explanation: it defines a binary optimality variable $o \in \{0, 1\}$ through $p(o = 1 \mid x_1, c) = y$, where $o = 1$ denotes that x_1 is optimal among the samples for its prompt (Levine, 2018). In both objectives, $\epsilon \sim \mathcal{N}(0, I)$ noises the head's input at time $t \in [t_{\min}, t_{\max}]$, drawn independently of the training example and of the ϵ' used to form $\hat{x}_1$.

The optimal heads under binary cross entropy loss read:

$$D^*(x_t, c, t) = \log \frac{p_{\text{ref},t}(x_t \mid c)}{p^{\text{pred}}_{\theta,t}(x_t \mid c)}, \quad R^*(x_t, c, t) = \log \frac{p(o = 1 \mid x_t, c, t)}{p(o = 0 \mid x_t, c, t)}, \quad (7)$$

where $p_{\text{ref},t}$ and $p^{\text{pred}}_{\theta,t}$ denote the noisy distributions of reference images and of generator predictions after noising by $\mathcal{N}_t$, and $p(o = 1 \mid x_t, c, t)$ is the probability that the rollout endpoint noised into x_t is optimal. For D_ϕ, the target is 1 on references and 0 on predictions, so its conditional mean at x_t is the share of the density at x_t contributed by references; the corresponding log-odds is the log density ratio. For R_ψ, the target is y itself, so $\sigma(R^*) = \mathbb{E}[y \mid x_t, c, t] = p(o = 1 \mid x_t, c, t)$. In practice, both heads only approximate these targets, limited by the frozen backbone features and by optimization error.

Optimality reward labels. Following DiffusionNFT (Zheng et al., 2025), we build y from rewards normalized within each prompt group. For each prompt c we draw a group $\mathcal{G} = \{x^i\}_{i=1}^K$ of endpoints from the current generator, evaluate $r_i = r(x^i, c)$, and define

$$A_i = \frac{r_i - \bar{r}}{\text{Std}(r) + \delta}, \qquad y_i = \frac{1}{2} + \frac{1}{2}\,\text{clip}\left(\frac{A_i}{C}, -1, 1\right), \quad (8)$$

where $\bar{r}$ and $\text{Std}(r)$ are the group mean and standard deviation, $\delta > 0$ stabilizes the normalization, and $C > 0$ controls the labels' sharpness. For the i-th sample of $\mathcal{G}$, the optimality variable from above therefore satisfies $p(o_i = 1 \mid x^i, c, \mathcal{G}) = y_i$. The labels average to $\frac{1}{2}$ when no advantage is clipped.

3.3 Generator objective

With both heads fixed, we train the generator to maximize their weighted logits:

$$\begin{aligned} \mathcal{L}_f(\theta) &= \mathbb{E}_{c,\hat{x}_1,t,\epsilon}[-s(\mathcal{N}_t(\hat{x}_1; \epsilon), c, t)]\,, \\ s(x_t, c, t) &= \lambda_D D_\phi(x_t, c, t) + \lambda_R R_\psi(x_t, c, t), \end{aligned} \quad (9)$$

where $\lambda_D, \lambda_R \geq 0$, $\epsilon \sim \mathcal{N}(0, I)$, and t is sampled as in head training. The expectation averages over prompts, generator predictions, and noising randomness. During each generator update, the head parameters and sampled rollout inputs remain fixed, while gradients propagate through the heads and frozen backbone to the prediction $\hat{x}_1 = f_\theta(x_{t_i}, c, t_i)$.

[1] Pairwise preference data can be incorporated directly by assigning target 1 to the winner and 0 to the loser, yielding the same binary classification loss form as the reference head.

Algorithm 1 DualHeads

Require: prompts with paired reference images; frozen backbone F; generator f_θ; heads D_ϕ, R_ψ; reward r; λ_D, λ_R, C, group size K

1: For each prompt c, run the sampler Eq. (2) K times; score the endpoints x_1 with r
2: Form soft labels y from group-relative advantages ▷ Eq. (8)
3: Renoise each x_1 to a grid time t_i; recompute $\hat{x}_1 = f_\theta(x_{t_i}, c, t_i)$
4: Update ϕ on $\mathcal{L}_D$ with x_{ref} against $\hat{x}_1$, and ψ on $\mathcal{L}_R$ with x_1 and y ▷ Eqs. (5) and (6)
5: Update θ by ascending $s(\mathcal{N}_t(\hat{x}_1; \epsilon), c, t)$ through $\hat{x}_1$, heads fixed ▷ Eq. (9)

Reverse-KL interpretation. Given an optimal reference head, Eq. (9) reduces to a standard KL-regularized reward objective. Fix a prompt c, a noise level t and the reward head. Since D^* in Eq. (7) is a log density ratio, $\mathbb{E}_{p^{\text{pred}}_{\theta,t}}[D^*] = -\text{KL}(p^{\text{pred}}_{\theta,t} \,\|\, p_{\text{ref},t})$, and substituting it into Eq. (9) gives

$$\mathcal{L}^*_{f,c,t}(\theta) = \lambda_D \,\text{KL}\Big(p^{\text{pred}}_{\theta,t}(\cdot \mid c) \,\Big\|\, p_{\text{ref},t}(\cdot \mid c)\Big) - \lambda_R \,\mathbb{E}_{x_t \sim p^{\text{pred}}_{\theta,t}(\cdot|c)}[R_\psi(x_t, c, t)]\,. \tag{10}$$

The first reference term penalizes the reverse KL to the reference distribution, while the second reward term favors large reward-head logits.

Connection to reward tilting. For $\lambda_D, \lambda_R > 0$, Eq. (10) is the standard KL-regularized objective (Schulman et al., 2017; Ziegler et al., 2019; Ouyang et al., 2022), whose minimizer is

$$p^*_t(x_t \mid c) \propto p_{\text{ref},t}(x_t \mid c) \exp\left(\frac{\lambda_R}{\lambda_D} R_\psi(x_t, c, t)\right), \tag{11}$$

which has the same form as Eq. (4), with R_ψ in place of r and λ_D/λ_R in place of β. The ratio of the head weights therefore sets the regularization strength. Note that Eq. (11) holds for the noised predictions at each t, and that R_ψ is fitted on samples from the current generator, so p^*_t is a fixed point of training. For unclipped labels close to $\frac{1}{2}$, $R^* \approx \frac{2}{C}\,\mathbb{E}[A \mid x_t, c, t]$, which gives the equivalent temperature $\beta_{\text{eq}} = C\lambda_D/(2\lambda_R)$ on the scale of the normalized advantage in Eq. (8).

Training and inference. Training alternates between classifier and generator updates at each iteration (Algorithm 1). For each prompt in the batch, we sample K few-step rollouts and evaluate their black-box rewards. The resulting rewards are used to calculate the soft label and then supervise the reward head. The same rollouts are re-noised to intermediate generator timesteps and passed once through f_θ to obtain predictions $\hat{x}_1$; the reference head is trained to distinguish these predictions from the prompt-paired reference images. After updating both heads, we freeze them and update the generator through the weighted classifier logits (Fig. 2). After training, we discard both heads; hence, at inference, we only use the fine-tuned few-step generator, at no extra inference cost.

4 Experiments

4.1 Experimental setup

Models and rewards. We use SD3.5 Medium (SD3.5-M) and Z-Image as base models, trained at a resolution of 512^2. With SD3.5-M, we study two non-differentiable rewards, OCR and GenEval, trained separately, and one multi-reward setting, PickScore + CLIPScore + HPSv2.1. For Z-Image, we use HPSv3 as reward. The generators are initialized as few-step distilled diffusion models.

Prompt dataset. For OCR and GenEval, we use the corresponding training and test sets from FlowGRPO. For all other rewards, we train on Pick-a-Pic (Kirstain et al., 2023) and evaluate on DrawBench (Saharia et al., 2022). Regardless of the training reward, we additionally evaluate every trained model on the OCR and GenEval test sets using their respective rewards.

Reference pools. For OCR and GenEval, we generate reference pools with the base model using training-set prompts and 20 steps of sampling. For OCR, we generate eight candidates for each of 5,000 prompts and retain the highest-scoring candidate for each prompt only if its OCR reward is at least 0.6. For GenEval, we construct a separate pool using 10,000 rows sampled from the GenEval

Table 1: Comparison on compositional image generation (GenEval), visual text rendering (OCR), and human preference metrics. Image quality and preference scores are computed on DrawBench (Saharia et al., 2022) prompts. Baseline numbers are from Flow-GRPO (Liu et al., 2025) where reported, and from our reimplementation otherwise. Abbreviations: Aes. (Aesthetics), ImgRwd (ImageReward), Pick (PickScore), CLIP (CLIPScore), HPS (HPSv2.1). NFE: number of function evaluations. Few-step RL baselines are quoted from their papers; RTDMD reports only GenEval for its GenEval-only run.

	Method	NFE	GenEval↑	OCR↑	Aes.↑	ImgRwd↑	Pick↑	CLIP↑	HPS↑
			Task Metric		Held-out Metric				
	SD3.5-M (base)	80	0.63	0.59	5.39	0.87	22.34	27.99	0.279
	Compositional image generation								
Multi-step RL	Flow-GRPO (Liu et al., 2025) w/o KL	80	0.95	—	4.93	0.44	21.16	—	—
Multi-step RL	Flow-GRPO (Liu et al., 2025) w/ KL	80	0.95	—	5.25	1.03	22.37	29.25	0.274
Multi-step RL	AWM (Xue et al., 2025a) w/ CFG	80	0.86	—	5.25	1.06	22.12	28.86	0.278
Multi-step RL	AWM (Xue et al., 2025a) w/o CFG	40	0.91	—	5.10	0.39	21.76	26.99	0.235
Multi-step RL	DiffusionNFT (Zheng et al., 2025)	40	0.92	—	5.30	0.63	22.07	27.25	0.253
Multi-step RL	CRD (Zhu et al., 2026)	40	0.93	—	5.44	0.98	22.48	28.44	0.284
Multi-step RL	RSM (Lee et al., 2026)	40	0.97	—	—	—	22.28	31.60	—
Few-step RL	TDM-R1 (Luo et al., 2026)	4	0.92	—	5.42	1.11	22.39	—	—
Few-step RL	RTDMD (Huang et al., 2026b)	4	0.94	—	—	—	—	—	—
Few-step RL	REST (Li et al., 2026b)‡	8	0.94	—	5.67	1.05	22.79	—	0.300
Few-step RL	**DualHeads**	2	0.89	—	5.53	1.01	22.42	28.56	0.281
Few-step RL	**DualHeads**	4	0.90	—	5.50	1.01	22.45	28.51	0.285
	Visual text rendering								
Multi-step RL	Flow-GRPO (Liu et al., 2025) w/o KL	80	—	0.93	5.13	0.58	21.79	—	—
Multi-step RL	Flow-GRPO (Liu et al., 2025) w/ KL	80	—	0.92	5.32	0.95	22.44	28.86	0.282
Multi-step RL	AWM (Xue et al., 2025a) w/ CFG	80	—	0.96	5.35	0.96	22.36	28.70	0.283
Multi-step RL	AWM (Xue et al., 2025a) w/o CFG	40	—	0.97	5.22	−0.64	20.82	24.54	0.203
Multi-step RL	DiffusionNFT (Zheng et al., 2025)	40	—	0.97	4.89	−0.81	20.53	24.08	0.196
Multi-step RL	CRD (Zhu et al., 2026)	40	—	0.92	5.33	0.87	22.40	28.48	0.281
Few-step RL	TDM-R1 (Luo et al., 2026)	4	—	0.95	5.45	1.09	22.62	—	—
Few-step RL	DMSampler (Liu et al., 2026)	4	—	0.83	—	—	—	—	—
Few-step RL	REST (Li et al., 2026b)‡	8	—	0.96	5.65	1.00	22.83	—	0.300
Few-step RL	**DualHeads**	2	—	0.91	5.41	1.06	22.50	29.06	0.293
Few-step RL	**DualHeads**	4	—	0.92	5.46	1.10	22.65	28.75	0.300
Few-step RL	**DualHeads** + Nano Banana pool	4	—	0.92	5.62	1.08	22.59	26.86	0.298

‡REST is trained jointly on the task reward and PickScore.

training set. For all other rewards, we use the existing Pick-a-Pic images as the reference pool. Alternatively, an external model can generate the reference pool from the same training prompts: we use Nano Banana, Google's proprietary image-generation model.

Training and evaluation. Following prior work, we fine-tune the student generator using LoRA with rank 32 and $\alpha = 64$. Each epoch consists of 48 prompt groups, with group size K=24 for the main results and K=8 for all ablations. By default, the reward head is warmed up on samples from the reference distribution with scores from the reward, so that the generator does not receive noisy gradients early in training. This warm-up uses 1000 steps, a very small training budge compared with a full run (Section 4.3). Both training and evaluation use four sampling steps with LCM sampler (Luo et al., 2023) and no classifier-free guidance (CFG scale 1). We also evaluate the trained models using Aesthetic and ImageReward on the same evaluation set. Details in Section B.

4.2 Task Performance

Text rendering. With four steps, DualHeads raises OCR accuracy from 0.59 to 0.92 (0.91 with two), on par with Flow-GRPO with KL and CRD (Table 1). AWM without CFG and DiffusionNFT reach 0.97 but lose image quality (negative ImageReward; HPS $\approx$ 0.20 vs. 0.279 for the base model). DualHeads instead improves all quality and preference scores over the base model, with the highest ImageReward (1.10) and HPS (0.300, tied with REST) among OCR-trained models. Among few-step methods, only TDM-R1 (0.95) and REST (0.96) score higher, using the teacher's score and 8 NFE with an extra PickScore reward, respectively.



Figure 3: **Qualitative comparison on OCR and GenEval test prompts.** DualHeads + NB signifies training with the Nano Banana image pool.

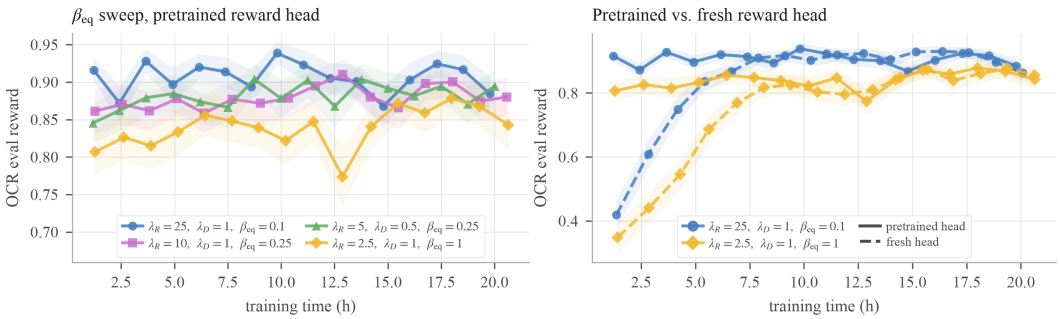
Table 2: **Reinforcing Z-Image** (Z-Image Team et al., 2025) with `HPSv3` (Ma et al., 2025) as the training reward (*). Z-Image-Turbo and TDM-R1 (Luo et al., 2026) are evaluated at 1024^2, except the entries marked †, which are reported in Table 3 of TDM-R1 (at 512^2, with Z-Image-Turbo at 4 NFE); the DualHeads rows are EMA weights of one run (reward weight 25) after 120 training iterations, sampled with two and four steps. `NFE`: number of function evaluations; `GenEval`: official overall accuracy; `OCR`: mean OCR score; `ImgRwd`: ImageReward; `CLIP`: CLIPScore. The best entry per column is in **bold**; shaded rows denote our method.

		Verifiable metrics		*Model-based metrics*					
Model	NFE	GenEval↑	OCR↑	HPSv3*↑	Aes.↑	ImgRwd↑	Pick↑	CLIP↑	HPSv2.1↑
Z-Image (base; Z-Image Team et al., 2025)	100	0.66†	0.74†	9.93	5.36	0.92	22.39	28.47	0.285
Z-Image-Turbo (DMDR; Jiang et al., 2026)	4	0.73†	0.78†	11.23	**5.42**	0.96	22.86	27.42	0.296
TDM-R1 (Luo et al., 2026)	4	**0.77**†	**0.79**†	12.67	5.39	**1.16**	22.82	**28.52**	0.307
DualHeads	2	0.76	0.64	12.91	5.27	1.10	22.77	27.31	0.307
DualHeads	4	0.74	0.75	**13.93**	5.36	1.07	**23.03**	27.12	**0.314**

Compositional generation. On GenEval, DualHeads raises the overall score from 0.63 to 0.90, within the range of multi-step methods (0.86–0.97) and slightly below the few-step baselines (0.92–0.94), which again use the teacher's score or 8 NFE. Position (0.24 → 0.90) and counting (0.50 → 0.91) gain the most; attribute binding remains the weakest (0.76; Table 5 in Section C.2). DualHeads again improves all quality and preference scores over the base model, whereas Flow-GRPO without KL, AWM without CFG and DiffusionNFT lower ImageReward (0.39–0.63) and PickScore.

Qualitative comparison. In Fig. 3, DualHeads renders all five OCR texts correctly and blends them into the scene (neon signs, a trail sign, a mural, an envelope). MeanFlowNFT prints the text on flat black plates and misspells "Upgrades Available", RTDMD mirrors or misspells words ("Propety", "Readiing"), and TDM-R1 adds a word ("Clup"). On GenEval, the baselines draw isolated objects on plain backgrounds (RSM also misses the TV), while DualHeads often places them in context, e.g., a ski in snow or a zebra beside a bed. The Nano Banana pool makes samples more photographic; it keeps OCR at 0.92 but lowers GenEval to 0.73 (Section 4.3).

Model-based reward on image quality. On Z-Image-Turbo with HPSv3 as the reward (Table 2), four-step DualHeads raises HPSv3 from 11.23 to 13.93, above TDM-R1 (12.67) and the 100-NFE base model (9.93). The gain carries over to held-out preference models, with the best PickScore (23.03) and HPSv2.1 (0.314) and a higher ImageReward (0.96 → 1.07), while GenEval and OCR stay close to Z-Image-Turbo. Aesthetic and CLIPScore drop slightly, and TDM-R1 keeps a higher ImageReward and CLIPScore (samples in Fig. 8). With a PickScore + CLIPScore + HPSv2.1 reward on SD3.5-M, DualHeads improves all five metrics over the base model but trails MeanFlowNFT and RTDMD on most (Table 4 in Section C.1).



(a) β_{eq} sweep, warmed-up reward head.

(b) Warmed-up vs. fresh reward head.

Figure 4: **Ablations on OCR.** Evaluation OCR score during training, (a) for several values of β_{eq} with a warmed-up reward head, and (b) for a warmed-up versus a freshly initialized reward head.

4.3 Ablations

Reward strength: Sensitivity to $\beta_{\text{eq}} \propto \lambda_R/\lambda_D$. Since $\beta_{\text{eq}} \propto C\,\lambda_D/\lambda_R$ with $C = 2.5$ fixed, sweeping β_{eq} is exactly sweeping the reward-to-reference weight ratio, smaller β_{eq} meaning a more reward-weighted objective. OCR reward improves monotonically as β_{eq} falls (Fig. 4a): over the final four checkpoints, 0.907 at $\beta_{\text{eq}} = 0.1$, 0.888 at 0.25 and 0.862 at 1, against a ± 0.029 confidence interval. The ratio being the right parameter is confirmed by the two $\beta_{\text{eq}} = 0.25$ runs, which halve the absolute weights from $(\lambda_R, \lambda_D) = (10, 1)$ to $(5, 0.5)$ and still agree to 0.015 across all sixteen checkpoints. We see no collapse even at $\beta_{\text{eq}} = 0.1$, so the optimum may lie lower still.

Warming up the reward head buys speed, not quality. At two points of the sweep we initialize the reward head either from the warmed-up head used by default (Section 4.1) or from scratch, holding everything else fixed (Fig. 4b). A fresh head starts far behind: 0.497 and 0.459 below its warmed-up twin at the first checkpoint, as the generator receives noisy scoring signals before the reward head is well-trained. It recovers by step 3000, some 8 of the ≈ 20 wall-clock hours per run, and converges to within 0.003 and 0.004 of the warmed-up runs, far inside the ± 0.029 and ± 0.038 intervals. As this holds at both weight settings, the reward head warm-up only speeds up training, at a very small cost, and does not change the final result.

Choice of data source for the reference pool. For the same training prompts, we compare a *base-model pool*, sampled from the base model, with a *Nano Banana pool* generated through its API. The two differ well beyond their scores: base-model samples keep SD3.5-M's flat lighting, sparse layouts and misspelled words, whereas Nano Banana images are photographic, correctly lettered and set in fuller scenes. On OCR, the generator adopts the pool's style without losing the text (last two rows of Fig. 3; more samples in Fig. 7): OCR stays at 0.92 (Table 1), the aesthetic score rises from 5.46 to 5.62, ImageReward, PickScore and HPS barely change, and CLIPScore drops from 28.75 to 26.86. GenEval, however, falls from 0.90 to 0.73, showing that the pool can also affect compositional accuracy. The two heads thus largely divide the work: the reward head governs what the reward measures, and the reference head governs everything the reward leaves unspecified, which is set by the reference pool.

5 Conclusion

We presented DualHeads, a simple method for adapting few-step diffusion generators using only external reward evaluations and reference samples. It turns KL-regularized reward maximization into two binary classification problems, solved by lightweight heads on a frozen diffusion backbone. With four steps, it reaches OCR and GenEval scores of 0.92 and 0.90, comparable to RL methods with 40-80 NFE, while improving all quality and preference scores over the base model, and gives the best HPSv3 on Z-Image-Turbo. The two heads divide the work: the reward head sets what the reward measures, and the reference pool largely sets the rest, such as style. Section A discusses limitations and future directions.

AI USE STATEMENT

Generative AI tools were used in several parts of this work: to help structure the paper and draft and edit its text and equations, to search and summarize related literature, to inspect implementation and experiment records, and to suggest experimental controls and interpretations of results. The authors reviewed all of this material before using it; in particular, citations and literature summaries were checked against the original papers, and all reported numbers were checked against the experiment logs. The authors take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Section 3 specifies the method, and Algorithm 1 summarizes one training iteration. Section 4.1 describes the base models, rewards, prompt sets, reference pools, and evaluation protocol, and Section B gives the training cycle, head architecture, update schedule, and hyperparameters. The base models, prompt sets, and reward models we use are publicly available, and we will release the code, configuration files, and reference pools upon publication.

REFERENCES

Andreas Bergmeister, Stefanie Jegelka, Nikolas Nüsken, Carles Domingo-Enrich, and Jakiw Pidstrigach. Reinforce adjoint matching: Scaling rl post-training of diffusion and flow-matching models. *arXiv preprint arXiv:2605.10759*, 2026.

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

Black Forest Labs. Flux. https://blackforestlabs.ai/announcing-black-forest-labs/, August 2024.

Guanjie Chen, Shirui Huang, Yifu Sun, Kai Liu, Jianchen Zhu, Xiaoye Qu, Yu Cheng, and Peng Chen. Flash-dmd: Towards high-fidelity few-step image generation with efficient distillation and joint reinforcement learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6010–6020, 2026.

Jiaxiang Cheng, Bing Ma, Xuhua Ren, Kai Yu, Peng Zhang, Tianxiang Zheng, and Qinglin Lu. Reward lightning: Fast video generation via homologous preference distillation. *arXiv preprint arXiv:2607.03960*, 2026.

Jaemoo Choi, Wei Guo, Yuchen Zhu, Arash Vahdat, Molei Tao, Julius Berner, and Yongxin Chen. Scaling reinforcement learning for diffusion models via velocity matching. *arXiv preprint arXiv:2608.23664*, 2026a.

Jaemoo Choi, Yuchen Zhu, Wei Guo, Petr Molodyk, Bo Yuan, Jinbin Bai, Yi Xin, Molei Tao, and Yongxin Chen. Rethinking the design space of reinforcement learning for diffusion models: On the importance of likelihood estimation beyond loss design. *arXiv preprint arXiv:2602.04663*, 2026b.

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

Linwei Dong, Ruoyu Guo, Ge Bai, Zehuan Yuan, Yawei Luo, and Changqing Zou. Guiding distribution matching distillation with gradient-based reinforcement learning. *arXiv preprint arXiv:2604.19009*, 2026.

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 *Forty-first International Conference on Machine Learning*, 2024.

Linqian Fan, Peiqin Sun, Tiancheng Wen, Shun Lu, and Chengru Song. R dm: Re-conceptualizing distribution matching as a reward for diffusion distillation. In *European Conference on Computer Vision*, pp. 418–434. Springer, 2026.

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

Zhen Fang, Wenxuan Huang, Yu Zeng, Yiming Zhao, Shuang Chen, Kaituo Feng, Yunlong Lin, Lin Chen, Zehui Chen, Shaosheng Cao, et al. Flow-opd: On-policy distillation for flow matching models. *arXiv preprint arXiv:2605.08063*, 2026.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36: 52132–52152, 2023.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*, 2021.

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

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *Advances in Neural Information Processing Systems*, 38:167283–167308, 2026a.

Yushi Huang, Xiangxin Zhou, Ruoyu Wang, Chi Zhang, Jun Zhang, and Tianyu Pang. Reinforcing few-step generators via reward-tilted distribution matching. *arXiv preprint arXiv:2605.26108*, 2026b.

Yushi Huang, Xiangxin Zhou, Jun Zhang, Liefeng Bo, and Tianyu Pang. Meanflownft: Bringing forward-process rl to average-velocity generators. *arXiv preprint arXiv:2607.15273*, 2026c.

Dengyang Jiang, Dongyang Liu, Zanyi Wang, Qilong Wu, Liuzhuozheng Li, Hengzhuang Li, Xin Jin, Changsheng Lu, Zhen Li, Mengmeng Wang, et al. Distribution matching distillation meets reinforcement learning. In *European Conference on Computer Vision*, pp. 281–299. Springer, 2026.

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

Jeongjae Lee, Jinho Chang, Jeongsol Kim, and Jong Chul Ye. Reward score matching: Unifying reward-based fine-tuning for flow and diffusion models. *arXiv preprint arXiv:2604.17415*, 2026.

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.

Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

Quanhao Li, Junqiu Yu, Kaixun Jiang, Yujie Wei, Zhen Xing, Pandeng Li, Ruihang Chu, Shiwei Zhang, Yu Liu, and Zuxuan Wu. Diffusionopd: A unified perspective of on-policy distillation in diffusion models. *arXiv preprint arXiv:2605.15055*, 2026a.

Yuhan Li, Fangao Zeng, Sicong Kang, Mengfei Xu, Hao Zhou, Wei Li, Pipei Huang, and Bingbing Ni. Rl-native distillation: Exploiting scored trajectories for few-step image generation. *arXiv preprint arXiv:2608.09226*, 2026b.

Zhiqi Li, Wen Zhang, and Bo Zhu. Flow-map grpo: Reinforcement learning for few-step flow-map generators via anchored stochastic composition. *arXiv preprint arXiv:2607.00535*, 2026c.

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

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online rl. *arXiv preprint arXiv:2505.05470*, 2025.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Zunxu Liu, Aiqiu Wu, Zhaofan Qiu, Yingwei Pan, Ting Yao, and Tao Mei. Distillation models are good samplers for diffusion reinforcement learning. In *Forty-third International Conference on Machine Learning*, 2026.

Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023.

Weijian Luo. Diff-instruct++: Training one-step text-to-image generator model to align with human preferences. *arXiv preprint arXiv:2410.18881*, 2024.

Yihong Luo, Tianyang Hu, Jiacheng Sun, Yujun Cai, and Jing Tang. Learning few-step diffusion models by trajectory distribution matching. *arXiv preprint arXiv:2503.06674*, 2025a.

Yihong Luo, Tianyang Hu, and Jing Tang. Reinforcing diffusion models by direct group preference optimization. *arXiv preprint arXiv:2510.08425*, 2025b.

Yihong Luo, Tianyang Hu, Weijian Luo, and Jing Tang. Tdm-r1: Reinforcing few-step diffusion models with non-differentiable reward. *arXiv preprint arXiv:2603.07700*, 2026.

Yuhang Ma, Xiaoshi Wu, Keqiang Sun, and Hongsheng Li. HPSv3: Towards wide-spectrum human preference score. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15086–15095, 2025.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

Mihir Prabhudesai, Anirudh Goyal, Deepak Pathak, and Katerina Fragkiadaki. Aligning text-to-image diffusion models with reward backpropagation. 2023.

Mihir Prabhudesai, Russell Mendonca, Zheyang Qin, Katerina Fragkiadaki, and Deepak Pathak. Video diffusion alignment via reward gradients. *arXiv preprint arXiv:2407.08737*, 2024.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

Axel Sauer, Frederic Boesel, Tim Dockhorn, Andreas Blattmann, Patrick Esser, and Robin Rombach. Fast high-resolution image synthesis with latent adversarial diffusion distillation. In *SIGGRAPH Asia 2024 Conference Papers*, pp. 1–11, 2024a.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *European Conference on Computer Vision*, pp. 87–103. Springer, 2024b.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2020.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International conference on machine learning*. PMLR, 2023.

Xu Wang, Zexian Li, Litong Gong, Tiezheng Ge, and Zhijie Deng. Advdmd: Adversarial reward meets dmd for high-quality few-step generation. *arXiv e-prints*, pp. arXiv–2604, 2026.

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.

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

Shuchen Xue, Chongjian Ge, Shilong Zhang, Yichen Li, and Zhi-Ming Ma. Advantage weighted matching: Aligning rl with pretraining in diffusion models. *arXiv preprint arXiv:2509.25050*, 2025a.

Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, et al. Dancegrpo: Unleashing grpo on visual generation. *arXiv preprint arXiv:2505.07818*, 2025b.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and Bill Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in neural information processing systems*, 37:47455–47487, 2024a.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6613–6623, 2024b.

Tianwei Yin, Qiang Zhang, Richard Zhang, William T Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast autoregressive video diffusion models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22963–22974. IEEE, 2025.

Z-Image Team, Huanqia Cai, Sihan Cao, Ruoyi Du, Peng Gao, Aiming Hao, Steven Hoi, Zhaohui Hou, Shijie Huang, Dengyang Jiang, Yuming Jiang, Xin Jin, Liangchen Li, Zhen Li, Zhong-Yu Li, David Liu, Dongyang Liu, Qilong Wu, Feng Yu, Zechao Zhan, Chi Zhang, Shifeng Zhang, Ruikai Zhou, and Shilin Zhou. Z-Image: An efficient image generation foundation model with single-stream diffusion transformer. *arXiv preprint arXiv:2511.22699*, 2025.

Shiyi Zhang, Mushui Liu, Yunze Tong, Wanggui He, Siyu Zou, Jinlong Liu, Yunlong Yu, Jian Song, Hao Jiang, Pipei Huang, et al. Self-opd: On-policy distillation for flow matching models without teacher. *arXiv preprint arXiv:2608.26872*, 2026.

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. Diffusionnft: Online diffusion reinforcement with forward process. *arXiv preprint arXiv:2509.16117*, 2025.

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 *Forty-first International Conference on Machine Learning*, 2024.

Yuanzhi Zhu, Xi Wang, Stéphane Lathuilière, and Vicky Kalogeiton. Diffusion reinforcement learning via centered reward distillation. *arXiv preprint arXiv:2603.14128*, 2026.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

Appendix for DualHeads

This appendix is organized as follows:

- Section A: Limitations and future work.
- Section B: Experimental details.
- Section C: Additional results and visualizations.

A Limitations and future work

We see three main limitations. **Idealized analysis.** Our analysis in Section 3.3 assumes optimal heads and a fixed reward head. In practice, the heads are only approximately optimal, and the reward head is refit on samples from the current generator, so the target moves during training. We also have no convergence guarantee for the alternating updates. **Learned reward.** The generator is trained against the reward head, not the reward itself, so it inherits both the errors of the head and the flaws of the external reward. The reference head keeps samples close to the reference pool, but it cannot tell when the generator is exploiting the reward. **Reference pool.** Fidelity is measured against the pool, so the generator also picks up properties that the reward ignores, such as style and composition (Section 4.3).

Several directions remain open. A convergence analysis of the alternating updates would put the method on firmer ground. The reward head also deserves a wider study, covering its architecture design, its backbone features analysis, the noise levels it sees, and so on. Richer reference pools, with several references per prompt or with real images, are another natural extension. Practically, DualHeads could be applied during distillation, at higher resolutions, and to few-step video generators.

This section extends the discussion of related work in Section 2.

B Experimental details

This section describes one training cycle, the head architecture, the update schedule, the rewards, and the reference pools; Section 4.1 lists the base models and prompt sets. Values in the text are those of the SD3.5-M OCR configuration; Table 3 lists the configuration for each reward. We use EMA model for inference across the experiments. The ablations use one NVIDIA H100 GPU and the main runs use 4.

B.1 Training cycle

Scored rollout endpoints. For each prompt the generator runs the sampler Eq. (2) without gradient, with $N = 4$ steps on the shifted time grid $(t_0, t_1, t_2, t_3) = (0, 0.1, 0.25, 0.5)$. The endpoints x_1 are decoded, scored with the reward r, and assigned the labels y of Eq. (8) with $C = 2.5$. The reward head is trained on x_1 renoised to noise level $\sigma = 1 - t \sim \mathcal{U}[0.25, 0.98]$, that is, $t \in [t_{\min}, t_{\max}] = [0.02, 0.75]$, with the image-level label y repeated over the patch map.

Selected reference images. Each prompt is paired with at most one reference latent from the pool (Section 4.1), repeated for every sample in its group. The reference image is the positive example of the reference head and is renoised over the same range $[t_{\min}, t_{\max}]$.

Recomputed generator predictions. An endpoint x_1 is renoised to one rollout grid timestep t_i, drawn once per group, giving $x_{t_i} = \mathcal{N}_{t_i}(x_1; \epsilon')$, and a single generator evaluation gives $\hat{x}_1 = f_\theta(x_{t_i}, c, t_i)$. A no-gradient copy of $\hat{x}_1$, renoised to $t \in [t_{\min}, t_{\max}]$, is the negative example of the reference head. The gradient copy, renoised to an independent t, is scored by both heads to form the utility s, and the generator update is the pathwise gradient of s through this single evaluation.

Table 3: Training configuration of the main runs for each reward.

		OCR	GenEval	Multi-reward	HPSv3
Model	Base model	SD3.5-M	SD3.5-M	SD3.5-M	Z-Image
	LoRA rank / α			32 / 64	
Sampling	Resolution			512^2	
	Sampler; steps; CFG			LCM; 4; 1	
	Time grid			$(0, 0.1, 0.25, 0.5)$	
Rollouts	Group size K			24	
	Prompt groups per epoch			48	
	Label sharpness C	2.5	2.5	2.5 / 0.5	2.5
	Head noise level $\sigma = 1 - t$			$[0.25, 0.98]$	
Reference pool	Source	5k OCR rows	10k GenEval rows	Pick-a-Pic images	Pick-a-Pic images
Heads	Levels; widths	$\{6, 12, 18, 22\}$; $1536 \to 640 \to 320$			$\{7, 14, 21, 29\}$; $3840 \to 640 \to 320$
	Parameters (both heads)	92.5M			208.3M
Optimization	Generator	AdamW, lr 2.5×10^{-5}, $(\beta_1, \beta_2) = (0.9, 0.999)$, weight decay 10^{-4}, grad. clip 1			
	Heads	AdamW, lr 10^{-4}, $(\beta_1, \beta_2) = (0.9, 0.999)$, weight decay 10^{-2}, no grad. clip			
	EMA decay d_k	$\min\{(1+k)/(10+k), 0.99\}$			
Utility	(λ_D, λ_R)	(1,10)	(1,50)	(1,25)	(1,25)
Precision		BF16 autocast; FP32 parameters, optimizer states, and VAE			

B.2 Heads details

Take SD3.5-M for example, both heads read the hidden states of four blocks, with zero-based indices $\mathcal{T} = \{6, 12, 18, 22\}$, of the frozen teacher transformer, reshaped to the 32×32 latent-patch grid at 512×512 resolution. Each level $k \in \mathcal{T}$ has its own convolutional head: 3×3 convolutions of widths $1536 \to 640 \to 320$ with GroupNorm and FiLM modulation (Sauer et al., 2024a) for conditions (c, t), followed by a zero-initialized 1×1 output projection. The level logits are summed,

$$D_\phi(x_t, c, t) = \sum_{k \in \mathcal{T}} D_{\phi,k}\big(F_k(x_t, c, t), c, t\big), \tag{12}$$

and analogously for R_ψ, where F_k is the feature map of block k. The reward head is an independent copy of the reference head's architecture with its own optimizer. Each head outputs a 32×32 map of patch logits; both BCE losses and the utility average over patches.

B.3 Optimization and hyperparameters

Table 3 lists the training configuration for each reward. The generator keeps an EMA $\bar{\theta}_k = d_k \bar{\theta}_{k-1} + (1 - d_k)\theta_k$; training rollouts use the raw weights. The utility weights are set through β_{eq} (Section 3.3), which Section 4.3 sweeps, for instance with $(\lambda_D, \lambda_R) = (1, 10)$ and $(0.5, 5)$.

B.4 Rewards

OCR. Text is recognized with PaddleOCR. The target string w is the first double-quoted span of the prompt; prompts without a non-empty target are excluded before scoring. Recognized segments with positive confidence are concatenated, in the order returned by the OCR engine, into a string $\hat{w}$. Both strings are lowercased and ASCII spaces are removed; other whitespace, punctuation, and Unicode forms are left unchanged. The reward is

$$r_{\text{OCR}}(\hat{w}, w) = \begin{cases} 1, & \text{if } w \text{ is a substring of } \hat{w}, \\ \max\{0,\, 1 - d_{\text{Lev}}(\hat{w}, w)/|w|\}, & \text{otherwise,} \end{cases} \tag{13}$$

where d_{Lev} is the Levenshtein distance and $|w|$ the target length; recognition failures receive zero. The reward therefore tolerates extra text and ignores case and spacing.

GenEval. Trained on the GenEval training prompts from Flow-GRPO, with a reference pool of 10,000 rows sampled from the GenEval training set (Section 4.1). We score images with the GenEval evaluator (Ghosh et al., 2023), as implemented in the Flow-GRPO reward server (Liu et al., 2025): Mask2Former (Swin-S, trained on COCO) detects objects with confidence above 0.3 (0.9 for counting), CLIP ViT-L/14 (Radford et al., 2021) classifies their colors, and relative positions are read from box centroids. A requested object with count n and $\hat{n}$ detections earns the credit $1 - |n - \hat{n}|/n$ and, only if $\hat{n} = n$, a further credit for its color or position. The *dense* training reward is the mean of these credits, while the *strict* score is 1 only if every requirement is met exactly. Tables 1 and 5 report the official GenEval accuracy, averaged over its six task categories.

Table 4: Comparison on multi-reward training with PickScore, CLIPScore and HPSv2.1. Image quality and preference scores are computed on `DrawBench` (Saharia et al., 2022) prompts. Rows marked † are quoted from Table 1 of MeanFlowNFT (Huang et al., 2026c), and the row marked ‡ from Choi et al. (2026b); the other rows are our evaluations. Abbreviations: `Aes.` (Aesthetics), `ImgRwd` (ImageReward), `Pick` (PickScore), `CLIP` (CLIPScore), `HPS` (HPSv2.1).

	Method	NFE	Aes.↑	ImgRwd↑	Pick* ↑	CLIP* ↑	HPS* ↑
	SD3.5-M (base)	80	5.39	0.87	22.34	27.99	0.279
Multi-step	SD3.5-M w/o CFG†	80	5.15	−0.48	20.66	23.91	0.211
Multi-step	SD3.5-M w/ CFG†	80	5.38	0.93	22.46	28.80	0.282
Multi-step	Flow-GRPO (Liu et al., 2025) w/ CFG†	80	5.28	1.02	22.48	29.12	0.266
Multi-step	DiffusionNFT (Zheng et al., 2025) w/o CFG†	40	5.96	1.41	23.64	28.89	0.324
Multi-step	Choi et al. (2026b)‡	40	6.06	1.45	23.68	29.60	0.325
Multi-step	CRD (Zhu et al., 2026)	40	5.70	1.35	23.27	29.63	0.321
Few-step	DMD (Yin et al., 2024b)†	4	5.51	0.92	22.28	28.41	0.287
Few-step	RTDMD (Huang et al., 2026b)†	4	6.13	1.23	23.28	27.75	0.327
Few-step	R_{dm} (Fan et al., 2026)†	4	5.75	0.72	22.07	27.20	0.276
Few-step	DMD + DiffusionNFT†	4	5.38	0.72	21.96	28.43	0.271
Few-step	CDM + DiffusionNFT†	4	4.83	0.15	21.38	27.45	0.214
Few-step	AnyFlow + DiffusionNFT†	4	5.95	1.24	23.09	29.15	0.292
Few-step	MeanFlowNFT (Huang et al., 2026c)†	4	5.93	1.45	23.50	29.67	0.327
Few-step	**DualHeads**	4	5.74	1.22	22.76	28.18	0.320

Multi-reward (PickScore + CLIPScore + HPSv2.1). Trained on Pick-a-Pic prompts, with the existing Pick-a-Pic images as the reference pool (Section 4.1). The three scores are PickScore (Kirstain et al., 2023) (the `PickScore_v1` checkpoint, on a CLIP ViT-H/14 backbone), CLIPScore (Hessel et al., 2021) with OpenAI CLIP ViT-L/14 (Radford et al., 2021), and HPSv2.1 (Wu et al., 2023) (on a CLIP ViT-H/14 backbone). Following DiffusionNFT (Zheng et al., 2025), the training reward is the sum of the three scores with equal weights, and the labels of Eq. (8) are computed from this sum.

HPSv3 (Z-Image). The generator is initialized from Z-Image-Turbo (Jiang et al., 2026), the few-step distilled version of Z-Image (Z-Image Team et al., 2025), and fine-tuned with the same LoRA configuration as SD3.5-M (Section 4.1). The reward is HPSv3 (Ma et al., 2025), with the released checkpoint built on Qwen2-VL. As in the multi-reward setting, we train on Pick-a-Pic prompts with the Pick-a-Pic images as the reference pool, and evaluate on DrawBench prompts. The reward weight is $\lambda_R = 25$, and Table 2 reports EMA weights after 120 training iterations, sampled with two and four steps; all other settings follow Table 3.

C Additional Results

C.1 Multi-reward training

We train SD3.5-M jointly on PickScore, CLIPScore and HPSv2.1 with Pick-a-Pic prompts, using the Pick-a-Pic images as the reference pool (Section 4.1), and evaluate on DrawBench prompts; Table 4 reports the results.

C.2 Per-category GenEval results

Table 5 breaks down the `GenEval` scores of Table 1 by category.

C.3 Qualitative results

Effect of the reward weight. Fig. 5 shows samples from the reward-weight sweep of Fig. 4a, with $\lambda_D = 1$, $C = 2.5$, and $\lambda_R \in \{2.5, 5, 10, 25\}$ from top to bottom; each column is one OCR test prompt. At $\lambda_R = 2.5$ the text is often small or misspelled, for example "pona" and "Hyddrate Now". As λ_R grows, the text becomes larger and easier to read and these errors disappear, in line

Table 5: **`GenEval` results by category.** Baseline numbers are taken from Flow-GRPO (Liu et al., 2025) or the original papers, and reimplemented where unavailable. Among the RL-tuned models, the best score per column is in **bold** and the second best is underlined; the shaded row is our method. Obj.: object; Attr. Bind.: attribute binding.

Model	Single Obj.	Two Obj.	Counting	Colors	Position	Attr. Bind.	Overall
Diffusion and flow-matching models							
SD-XL (Podell et al., 2023)	0.98	0.74	0.39	0.85	0.15	0.23	0.55
FLUX.1 Dev (Black Forest Labs, 2024)	0.98	0.81	0.74	0.79	0.22	0.45	0.66
SD3.5-M (Esser et al., 2024)	0.98	0.78	0.50	0.81	0.24	0.52	0.63
TDM-SD3.5-M (Luo et al., 2025a) (4 NFE)	0.99	0.77	0.49	0.79	0.23	0.44	0.61
Multi-step RL (SD3.5-M backbone)							
Flow-GRPO (Liu et al., 2025)	**1.00**	0.99	0.95	**0.92**	**0.99**	0.86	0.95
AWM (Xue et al., 2025a) (w/o CFG)	**1.00**	0.98	0.95	0.77	0.78	0.65	0.91
DiffusionNFT (Zheng et al., 2025)	**1.00**	0.99	0.96	0.82	0.78	0.73	0.92
CRD (Zhu et al., 2026)	**1.00**	0.98	0.92	0.88	0.90	0.73	0.93
DMSampler (Liu et al., 2026)	**1.00**	**1.00**	**0.98**	0.90	**0.99**	0.86	**0.96**
Few-step RL (SD3.5-M backbone)							
TDM-R1 (Luo et al., 2026)	**1.00**	0.96	0.88	0.85	0.93	0.91	0.92
RTDMD (Huang et al., 2026b)	**1.00**	0.98	0.95	0.90	0.87	**0.94**	0.94
DualHeads	**1.00**	0.98	0.91	0.85	0.90	0.76	0.90



Figure 5: Visualization of samples from the ablation Fig. 4a.

with the higher held-out OCR reward in Fig. 4a. At $\lambda_R = 25$ the text also tends to dominate the image, often as a bold sign or sticker, while the layout of each scene stays similar across rows.

Two-step sampling. The generators are trained with four sampling steps, but they can also be sampled with two. Fig. 6 shows four-step and two-step samples on the same prompts, for the OCR generator (a) and the GenEval generator (b). With two steps, the generator still reaches a reward close to that of four steps (Table 1).

Effect of the reference pool. Fig. 7 shows that the reference pool sets the style of the generator but not the content it renders.

Comparison of Z-Image-based models. Fig. 8 shows samples from the Z-Image base model alongside its distilled and RL-trained variants.

Uncurated samples. Figs. 9 to 11 show DualHeads samples only, taken from the evaluation runs. Prompts are drawn uniformly without replacement.

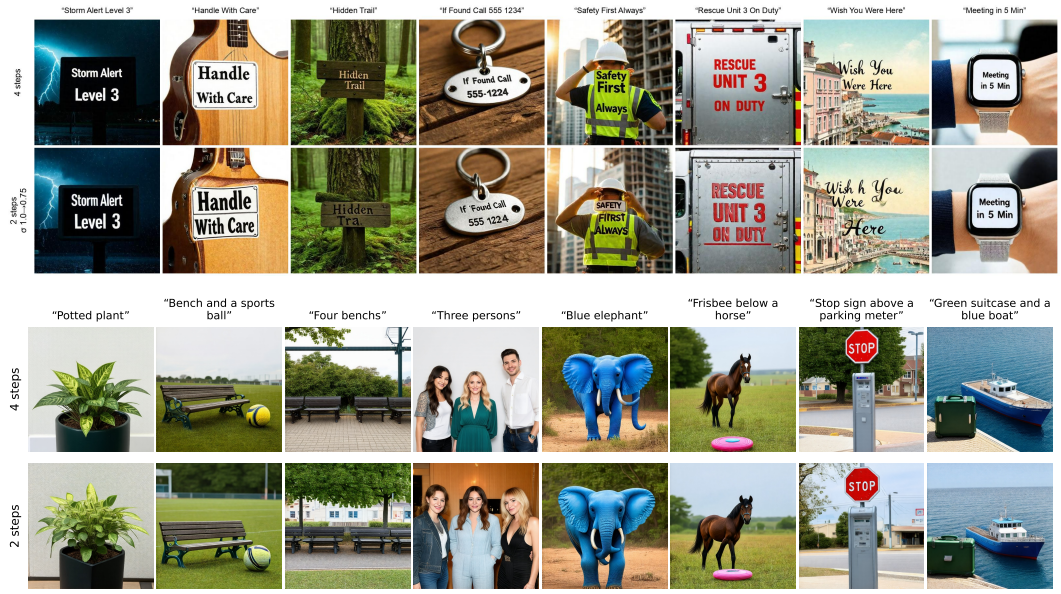


Figure 6: Visual comparison of four-step and two-step generation from the four-step generators trained with the OCR reward (top) and the GenEval reward (bottom).



Figure 7: **Effect of the reference pool.** Same prompts with teacher (top) and Nano Banana (bottom) references: the style changes, the content does not.

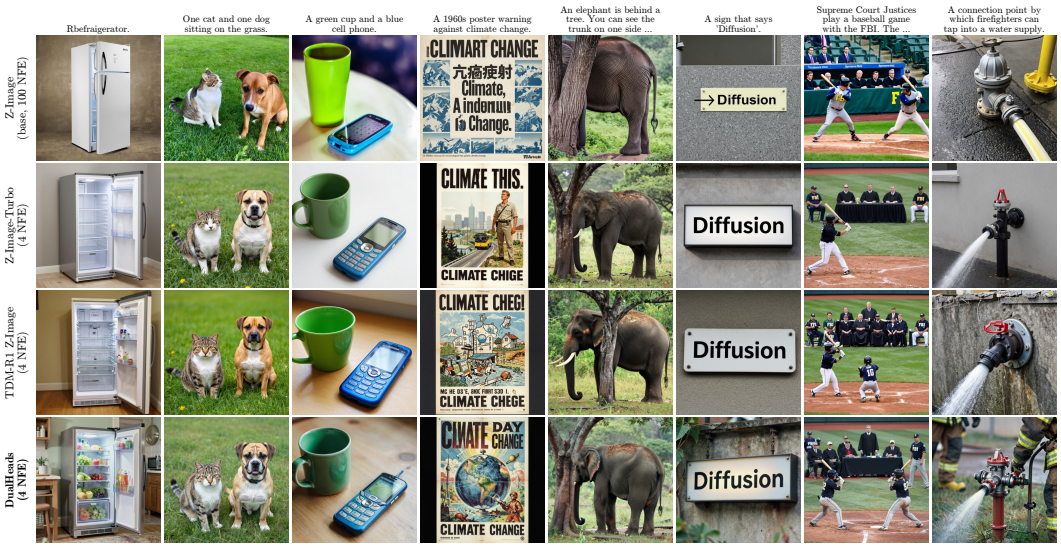


Figure 8: Qualitative comparison for Z-Image-based models.

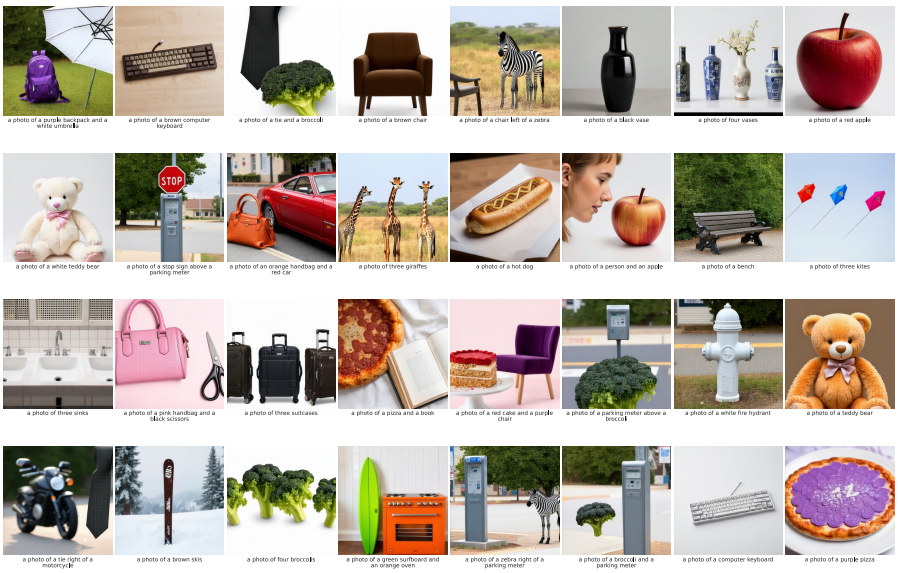


Figure 9: Uncurated samples on GenEval test prompts.

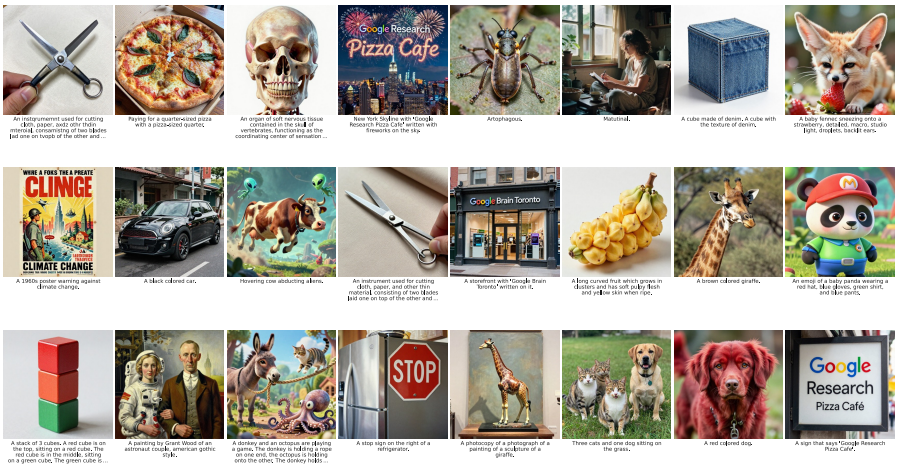


Figure 10: Uncurated Z-Image samples after HPSv3 training.

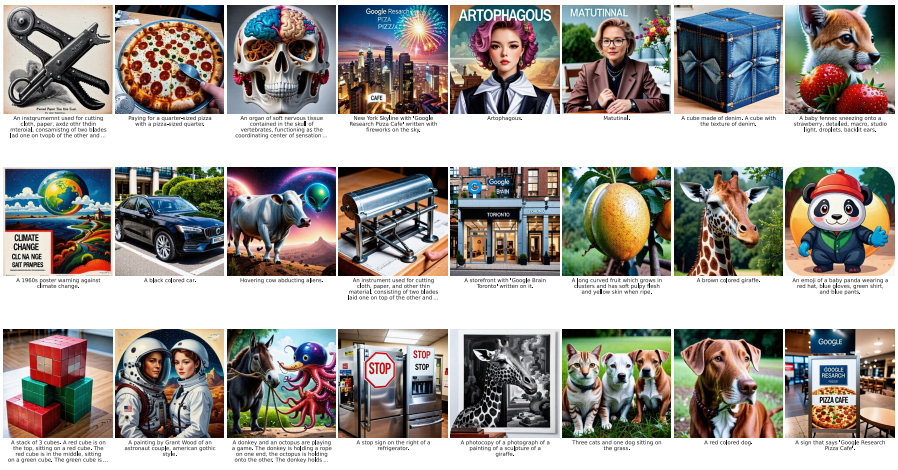


Figure 11: Uncurated samples on DrawBench prompts after multi-reward training (PickScore + CLIPScore + HPSv2.1).