REINFORCING DIFFUSION MODELS VIA VALUE-IMPLICIT REWARD MATCHING

Anonymous authors
Paper under double-blind review

ABSTRACT

Online Reinforcement Learning (RL) has proven effective for aligning diffusion and flow models with task-specific rewards. However, existing methods rely heavily on newly generated on-policy samples for stable policy improvement, leading to poor sample efficiency. We introduce **VIRM**, a value-implicit reward matching framework that casts policy improvement as regression on replayed samples. VIRM builds on the optimality condition of KL-regularized RL, where the optimal policy exponentially tilts the reference toward high reward. Inverting this identity reveals that each sample's reward is implicitly encoded by the policy–reference log-density ratio and a per-prompt log-partition term. Using an exponential moving average (EMA) policy as the reference, VIRM optimizes the current policy by matching this policy-implied reward to external feedback. Concretely, the log-density ratio is estimated via an ELBO difference that acts as an implicit value, while the intractable log-partition term is predicted by a token-wise value head. VIRM learns from replayed data without importance weights or stored denoising trajectories. To focus replay on correctable errors, samples are prioritized by reducible loss, a reference-calibrated measure of learnability rather than intrinsic difficulty. On GenEval2, VIRM improves FLUX.2-klein-base-9B from 25.4 to 65.4, outperforming the strongest baseline by more than 40%, with concurrent gains in CLIP score, PickScore, and HPSv2. These results establish value-implicit reward matching as a sample-efficient post-training method for diffusion and flow models.

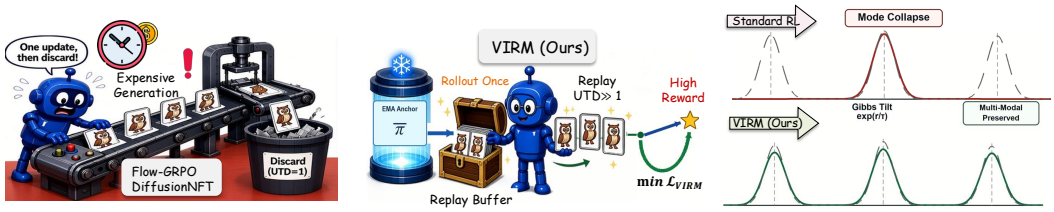


Figure 1: VIRM enable higher update-to-data (UTD) ratio via reward matching training pipeline, achieving superior sample efficiency than existing diffusion RL methods without mode collapse issues.

1 INTRODUCTION

Diffusion models, especially flow-matching models, have become a leading paradigm for high-fidelity generation in continuous domains, including images, videos, and audio (Lipman et al., 2022; Liu et al., 2023; Seedance et al., 2026; Brooks et al., 2024). Their scalability largely stems from simple supervised objectives that regress toward closed-form denoising or velocity targets (Song et al., 2020; Ho et al., 2020). Pretraining alone, however, cannot fully encode downstream requirements such as compositional correctness, text rendering, perceptual quality, or human preference. Online reinforcement learning (RL) (Schulman et al., 2017; Shao et al., 2024) addresses this gap by scoring generated outputs with external reward functions and steering the generative distribution toward the desired behavior. Recent methods have demonstrated substantial improvements across these objectives, establishing RL as an effective post-training approach for diffusion and flow models (Xue et al., 2025; Liu et al., 2025a; He et al., 2025c).

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.

These gains, however, come at a substantial data-acquisition cost. Trajectory-based methods stochasticize generation into an SDE trajectory and optimize its intermediate transitions (Ping et al., 2026), whereas forward-process methods construct training states by analytically re-noising generated outputs (Zheng et al., 2026; Xue et al., 2026; Bergmeister et al., 2026). Despite this difference, both rely heavily on newly generated rollouts to achieve stable improvements. Each rewarded output requires a full sampling pass and expensive evaluation by large vision-language models (Bai et al., 2025) or multiple task-specific verifiers, yet is typically used for only a single update before being discarded. This single-use training paradigm becomes the principal bottleneck to sample-efficient post-training.

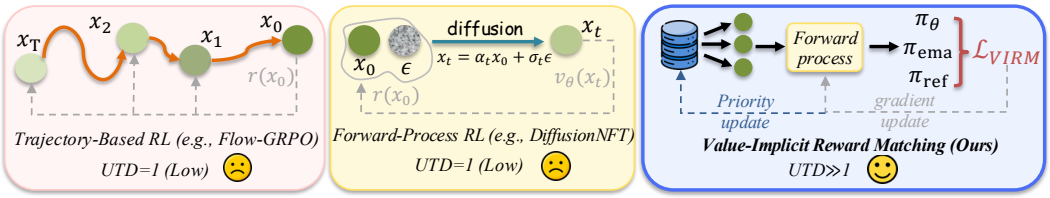


Figure 2: Existing diffusion RL methods obtain only a single update from each expensive rollout. VIRM instead stores outputs generated by a fixed EMA anchor and reuses them through reward matching, improving sample efficiency.

Naively replaying previously rewarded trajectories does not resolve this problem. Policy-gradient methods estimate an expectation under the current policy, so samples from earlier policies require an importance-weighting ratio to correct this distribution shift. For diffusion models, evaluating this ratio requires approximating an intractable likelihood. Moreover, importance ratios can become poorly conditioned as the current policy moves away from the behavior policy. Rather than reactively correcting old samples to approximate a changing on-policy gradient, we ask: *Is there a native off-policy RL algorithm that supports a high Update-To-Data (UTD) ratio?*

We answer this question with **Value-Implicit Reward Matching (VIRM)**, a replay-based post-training framework designed for high-UTD optimization. VIRM reformulates the training objective: instead of estimating an on-policy reward expectation, it reconstructs the implicit value encoded by the current policy and matches that value to external feedback. We view conditional generation as a one-step contextual bandit in which the terminal outcome is the action and the sampling path is internal computation. VIRM therefore reuses rewarded outputs without storing their generation trajectories or behavior-policy likelihoods, and remains agnostic to the sampler and its discretization.

The key insight comes from the Gibbs solution of KL-regularized policy improvement (Uehara et al., 2024). For a fixed EMA anchor $\bar{\pi}_{\text{ema}}$, the target policy $\pi^\star$ satisfies

$$\pi^\star(x_0 \mid c) = \frac{\bar{\pi}_{\text{ema}}(x_0 \mid c)\exp(r(x_0,c)/\tau)}{Z(c)} \quad \Longleftrightarrow \quad r(x_0,c) = \tau \log \frac{\pi^\star(x_0 \mid c)}{\bar{\pi}_{\text{ema}}(x_0 \mid c)} + \tau \log Z(c). \tag{1}$$

The log-density ratio $\tau \log(\pi_\theta(x_0 \mid c)/\bar{\pi}_{\text{ema}}(x_0 \mid c))$ measures how π_θ increases or decreases the probability of an output x_0 relative to the fixed anchor. $Z(c)$ serves as the normalizer. Together, these terms define the *policy-implied value* assigned to generation x_0. VIRM directly regresses this implied value toward the observed external reward, thereby transforming an MLE-style policy-gradient objective into an MSE-based regression loss. Because the anchor remains fixed within an interval, that target remains valid as π_θ changes, allowing previously generated outputs to be reused without requiring importance sampling ratios.

VIRM makes this idea practical with three components. First, group-wise reward centering removes offsets across groups without changing the reward-tilted distribution (Naik et al., 2024; Marek & Ryabinin, 2026). Second, an ELBO-based log-density-ratio surrogate uses the velocity-reconstruction errors to estimate $\log(\pi_\theta(x_0 \mid c)/\bar{\pi}_{\text{ema}}(x_0 \mid c))$. Third, we train an ensemble token-wise value head z_ϕ to approximate the normalizer correction $\tau \log Z(c)$ (Choi et al., 2026; Zhu et al., 2026) in the high-dimensional output space; unlike a conventional critic, it estimates a Gibbs normalization term rather than expected return.

To allocate the buffer budget effectively, VIRM introduces reducible-loss prioritized replay (Sujit et al., 2023), concentrating updates on correctable data rather than noise-dominated samples. Our theory shows that VIRM recovers the reward-tilted policy, with the gap to the optimal target bounded by the reward-matching residual and practical approximation errors.

We evaluate VIRM for post-training models across different architectures and scales, including SD3.5-Medium and FLUX.2-klein-base-9B (Esser et al., 2024; Labs, 2025). Performance is assessed on held-out datasets using a diverse set of metrics. On GenEval2 (Kamath et al., 2025), VIRM improves FLUX.2-klein-base-9B from 25.4 to 65.4, substantially outperforming the strongest baseline at 55.2. Our qualitative results further demonstrate that VIRM better preserves the diversity of the base model than conventional mode-seeking RL methods, producing a wider range of visually distinct outputs while maintaining strong semantic alignment and perceptual quality. We summarize our contributions as follows:

- We formulate policy improvement through policy-implied reward matching and establish a training pipeline that supports replay without importance weighting.
- We make reward matching tractable for diffusion and flow models through an ELBO-derived log-density-ratio surrogate, group-wise reward centering, and a learnable token-wise normalizer.
- We enable high-UTD training through anchor-consistent data reuse and reducible-loss prioritized replay, allowing each expensive rollout to support multiple policy updates.
- Our empirical results across multiple model architectures, datasets, and evaluation metrics highlight that VIRM consistently improves sample efficiency, generation quality, and diversity.

2 Related Work

Diffusion RL. Fine-tuning pretrained diffusion models with online reinforcement learning (RL) has demonstrated substantial improvement in aligning text-to-image generation with various downstream objectives. A large literature adapts Group Relative Policy Optimization (GRPO) (Shao et al., 2024) to flow models through ODE-to-SDE sampling (Liu et al., 2025a; Xue et al., 2025), with subsequent works targeting sample efficiency (Li et al., 2025a; Wang & Yu, 2025), update stability (He et al., 2025c; Wang et al., 2026; Li et al., 2025b), and credit assignment (He et al., 2025d; Deng et al., 2026; Savani et al., 2026; Lee & Ye, 2025). A complementary line of work instead leverages the forward process to train flow models, eliminating the need to estimate policy likelihood ratios while achieving competitive performance (Zheng et al., 2026; Xue et al., 2026; Luo et al., 2026; Bergmeister et al., 2026; Kveton et al., 2026). However, these methods permit only a single gradient update per rollout, resulting in poor sample efficiency given the high cost of sampling and reward evaluation. OP-GRPO (Zhang et al., 2026) addresses this problem within GRPO through reactive importance sampling corrections and conservative trust region safeguards. In contrast, VIRM converts clean rewarded outputs into anchor-consistent regression targets, enabling high-UTD replay without trajectory storage or current-to-behavior importance weighting under standard coverage conditions.

Reward Matching. Reward matching builds on the Gibbs solution of KL-regularized and maximum-entropy RL (Uehara et al., 2024). Closely related precedents include Path Consistency Learning, which fits soft value–policy identities along trajectories (Nachum et al., 2017), and GFlowNets, which use flow balance to make terminal sampling probabilities proportional to rewards (Bengio et al., 2023; He et al., 2025a). In contrast to these sequential formulations, VIRM treats conditional generation as a one-step contextual bandit and thus avoids consistency constraints over intermediate states. Recent work extends reward matching to diffusion and flow models through distribution matching or score estimation (Venkatraman et al., 2024; Lee et al., 2026; Zhu et al., 2026; Liu et al., 2025b), but focuses on which distribution to match rather than on how to reuse costly rewarded rollouts across many updates. VIRM addresses this by fixing the EMA anchor within each training interval, thereby preserving the validity of the matching target as generation data are reused, while prioritized replay allocates budget toward samples that remain learnable.

3 Preliminaries

Given a conditioning input $c \sim \mathcal{D}$ (e.g., a text prompt), flow matching (Lipman et al., 2022; Liu et al., 2023) corrupts a clean output x_0 as $x_t = (1-t)x_0 + t\epsilon$, with $\epsilon \sim \mathcal{N}(0, I)$ and $t \in [0, 1]$, and trains a velocity field $v_\theta(x_t, t, c)$ to regress $u^\star = \epsilon - x_0$. For trajectory-based methods, samples are generated by integrating an ODE or SDE from $t = 1$ to $t = 0$ (Kim et al., 2025; Singh & Fischer, 2024; He et al., 2025b). An SDE sampler on the grid $1 = t_T > \cdots > t_0 = 0$ defines an MDP with state $s_k = (c, t_k, x_{t_k})$, action $a_k = x_{t_{k-1}}$, policy $p_\theta(x_{t_{k-1}} \mid x_{t_k}, c)$, and a sparse reward $r(x_0, c)$.

Policy-gradient methods on this MDP (Liu et al., 2025a; Xue et al., 2025) therefore require stochastic transitions and stored trajectories with estimated per-step log-probabilities over training. Inspired by previous forward-process methods (Zheng et al., 2026; Xue et al., 2026; Luo et al., 2026), VIRM marginalizes the intermediate denoising chain, and further views the denoising sampling process as a one-step contextual bandit. Each rollout thus reduces to a tuple (c, x_0, r), where x_0 denotes the action. Our training objective is shown as follows under this formulation:

$$\max_{\pi} \mathcal{J}_c(\pi) := \mathbb{E}_{x_0\sim\pi(\cdot|c)}[r(x_0,c)] - \tau D_{\mathrm{KL}}\big(\pi(\cdot \mid c) \,\|\, \bar{\pi}_{\mathrm{ema}}(\cdot \mid c)\big), \quad \pi^\star(x_0 \mid c) = \frac{1}{Z(c)}\bar{\pi}_{\mathrm{ema}}(x_0 \mid c)\, e^{r(x_0,c)/\tau}. \tag{2}$$

The optimal policy exponentially tilts the reference toward high reward with temperature $\tau > 0$, and the per-prompt partition function $Z(c)$ normalizes the tilt. Flow-matching error on x_0 yields an evidence lower bound (ELBO) (Choi et al., 2026; Kingma & Gao, 2023):

$$\log \pi_\theta(x_0 \mid c) \geq C(x_0, c) - \mathbb{E}_t\big[w(t)\, \|v_\theta(x_t, t, c) - u^\star\|_2^2\big], \tag{3}$$

where $\mathbb{E}_t$ averages over t and ϵ, $w(t)$ is an ELBO-valid weight, and $C(x_0, c)$ is a θ-independent constant.

4 Method

We present *Value-Implicit Reward Matching* (VIRM), which learns a policy by matching external rewards with the value implicit in the policy. We first show our training objective and its equality-regression form, then introduce the practical estimators and stabilization mechanisms.

4.1 Value-Implicit Objective

Re-noising in previous forward-process methods makes rollout generation *evaluable*, but not *reusable*: advantage-weighted objectives such as AWM (Xue et al., 2026) and positive–negative splits such as DiffusionNFT (Zheng et al., 2026) define their optimum relative to the rollout distribution μ (e.g., maximizing $\mathbb{E}_{x_0\sim\mu}[g(x_0)\log\pi(x_0)]$ with $g \geq 0$ yields $\pi \propto \mu g$), so stale replay shifts their target. VIRM instead builds on the Gibbs solution, which constrains each sampled (c, x_0) separately. VIRM sets the reference $\bar{\pi}_{\mathrm{ema}}$ of Eq. (2) to an EMA copy of π_θ, which is fixed within each training interval, generates rollouts, and is updated at its end (see Appendix C), so each interval poses a fixed KL-regularized problem. Taking logarithms in Eq. (2) gives, whenever $\bar{\pi}_{\mathrm{ema}}(x_0 \mid c) > 0$,

$$r(x_0, c) = \tau \log Z(c) + \tau \log \frac{\pi^\star(x_0 \mid c)}{\bar{\pi}_{\mathrm{ema}}(x_0 \mid c)}. \tag{4}$$

A policy *implies* a reward for every generation x_0: the log-ratio measures how strongly the policy upweights x_0 relative to the anchor, and the log-normalizer $\tau \log Z(c)$ supplies the per-prompt offset. VIRM replaces $(\pi^\star, \tau \log Z)$ with a trainable pair (π_θ, z) and matches the implicit reward to the external reward feedback $r(x_0, c)$:

$$\widehat{r}_{\theta,z}(x_0, c) := z(c) + \tau \log \frac{\pi_\theta(x_0 \mid c)}{\bar{\pi}_{\mathrm{ema}}(x_0 \mid c)}, \qquad \mathcal{L}_{\mathrm{pop}}(\theta, z; \eta) := \mathbb{E}_{(c,x_0)\sim\eta}\big[(\widehat{r}_{\theta,z}(x_0, c) - r(x_0, c))^2\big], \tag{5}$$

where the data distribution η can differ from the current policy distribution. Because $\mathcal{L}_{\mathrm{pop}}$ regresses onto the pointwise identity in Eq. (4) rather than estimating an expectation under the current policy, the sampling distribution η only reweights the residuals and leaves the minimizer unchanged: $(\pi^\star, \tau \log Z)$ remains the unique optimal solution (Theorem 1). CRD (Zhu et al., 2026) applies the identity to diffusion models but cancels $\tau \log Z(c)$ within prompt groups, whereas VIRM keeps the per-output form, so a single output can be replayed at a high UTD ratio for improved sample efficiency.

4.2 Stabilized Practical Optimization

Group-Wise Reward Centering. Mean rewards vary widely across prompts, and the log-normalizer inherits this offset: by Jensen's inequality, $\tau \log Z(c) \geq b(c) := \mathbb{E}_{x_0\sim\bar{\pi}_{\mathrm{ema}}(\cdot|c)}[r(x_0, c)]$. With raw

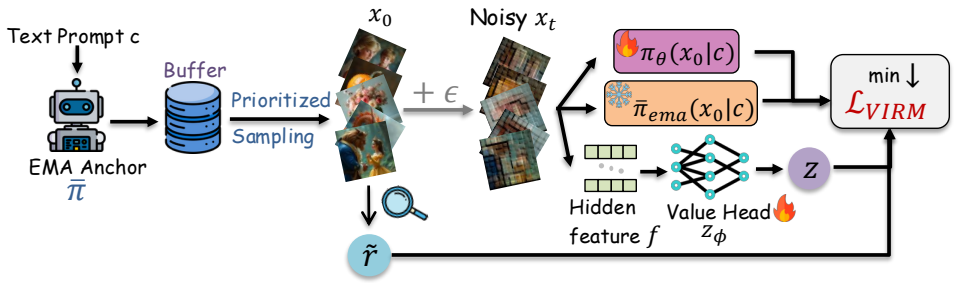


Figure 3: Overview of VIRM. Anchor rollouts are stored with their rewards and replayed by reducible-loss priority. The value head z_ϕ takes the hidden feature f as input and predicts the centered normalizer $z^\star(c)$ of Eq. (6) to compute $\mathcal{L}_{\text{VIRM}}$.

rewards, this between-prompt variation would dominate the regression and leak into the ratio term. VIRM therefore regresses the centered reward $r - b(c)$, for which Eq. (4) becomes

$$r(x_0, c) - b(c) = z^\star(c) + \tau \log \frac{\pi^\star(x_0 \mid c)}{\bar{\pi}_{\text{ema}}(x_0 \mid c)}. \tag{6}$$

The shift leaves $\pi^\star$ unchanged, and the centered normalizer $z^\star(c) := \tau \log Z(c) - b(c)$ reflects only how much rewards vary within a prompt (Appendix A.2). In practice, VIRM estimates $b(c)$ by the mean reward of the G rollouts of each prompt, so the regression target for output i is $\widetilde{r}_i := r_i - \frac{1}{G}\sum_{j=1}^{G} r(x_{0_j}, c_i)$. We denote the error of this estimate as ε_r in Theorem 2.

ELBO-Based Ratio Estimation. For a replayed output x_0, to estimate the intractable log-ratio $\log(\pi_\theta(x_0 \mid c)/\bar{\pi}_{\text{ema}}(x_0 \mid c))$, VIRM draws $t \sim \mathcal{U}[0,1]$ and $\epsilon \sim \mathcal{N}(0, I)$, sets $x_t = (1-t)x_0 + t\epsilon$ and $u^\star = \epsilon - x_0$:

$$q_{t,k} := -d\, w(t)\big[(v_\theta(x_t,t,c) - u^\star)_k^2 - (v_{\bar{\theta}_{\text{ema}}}(x_t,t,c) - u^\star)_k^2\big], \qquad \log \frac{\pi_\theta(x_0 \mid c)}{\bar{\pi}_{\text{ema}}(x_0 \mid c)} := \mathbb{E}_t\Big[\tfrac{1}{d}\textstyle\sum_{k=1}^{d} q_{t,k}\Big], \tag{7}$$

where k indexes the d elements of the velocity v_θ. Thus $\frac{1}{d}\sum_k q_{t,k} > 0$ when π_θ denoises x_0 better than the anchor, i.e., when it has moved probability toward x_0. ELBO differences are established are established surrogates for this log-ratio (Choi et al., 2026; Xue et al., 2026). Following Zheng et al. (2026), we use $w(t) = 1$ with $t \sim \mathcal{U}[0,1]$ for stability and simplicity. Theoretical analysis and more discussion of the ratio estimation are provided in Theorem 2 and Appendix B.1.

Token-Wise Value Head. The centered normalizer $z^\star(c)$ remains unknown. A log-mean-exp estimate over a small group is biased, and within-prompt centering (Bartoldson et al., 2025), as in CRD (Zhu et al., 2026), ties each output's loss to its group, so outputs cannot be replayed or prioritized individually. VIRM instead learns a head $z_\phi(x_t, t, c) \in \mathbb{R}^d$ on detached DiT features f, with one value per element of v_θ without pooling across tokens. This enables every internal token to receive its own reward target, i.e., $\widetilde{r}_i - z_{\phi,k}$, and these targets form a model-implied, pixel-wise reward map anchored to the image-level reward. Regressing only the average $\bar{z}_\phi := \frac{1}{d}\sum_{k=1}^{d} z_{\phi,k}$ would instead assign one advantage to all elements and restrict the policy to uniformly weighted updates (Appendix B.1). To improve the expressiveness of z_ϕ and reduce learning variance, VIRM averages a diverse ensemble of H heads in the forward pass (Peer et al., 2021; Liu et al., 2026) and back-propagates to a randomly sampled nonempty subset $S \subseteq [H]$ via the straight-through rule $z_\phi = \text{sg}(\langle z \rangle_{[H]}) + \langle z \rangle_S - \text{sg}(\langle z \rangle_S)$, with $\langle z \rangle_S$ the mean over S and sg the stop-gradient. Restricting updates to a random subset of heads reduces the correlation between their errors, and hence the variance of the full mean $\langle z \rangle_{[H]}$ used in the policy target.

Core Objective. Combining the three components, VIRM minimizes

$$\boxed{\mathcal{L}_{\text{VIRM}} := \mathbb{E}_{(i,t,\epsilon)\sim\nu}\Big[\tfrac{1}{d}\textstyle\sum_{k=1}^{d}\big(z_{\phi,k}(x_t,t,c_i) + \tau q_{t,k} - \widetilde{r}_i\big)^2\Big]} \tag{8}$$

Algorithm 1 provided in Appendix B summarizes the training pipeline.

4.3 Replay and Data Utilization

VIRM generates every output of an interval with the fixed anchor, stores only $(c_i, x_{0_i}, \widetilde{r}_i)$, and clears the buffer whenever the anchor is refreshed, so targets and ELBO comparisons never mix anchors.

Replay remains off-policy for the evolving π_θ, since $\mathcal{L}_{\mathrm{VIRM}}$ fits an η-independent equality rather than estimating an on-policy reward expectation.

Uniform replay can waste updates on useless samples, whereas using the absolute loss as replay priority can favor noisy or intrinsically difficult samples. VIRM instead prioritizes the reducible loss (Sujit et al., 2023): an output is learnable when training can still lower its loss.

Let $\mathcal{L}_{\theta,i}$ denote the per-sample VIRM loss and $\mathcal{L}_{\mathrm{ref},i}$ the reference loss obtained by replacing v_θ with the frozen pretrained v_{ref}. The priority update is defined as follows:

$$p_i = \mathrm{clip}\Big(\mathrm{sign}(\widetilde{r}_i)\big[\mathrm{sg}(\mathcal{L}_{\theta,i}) - \mathcal{L}_{\mathrm{ref},i}\big], 0, p_{\max}\Big), \qquad P(i) = \frac{(p_i + \varepsilon_p)^\alpha}{\sum_{i'}(p_{i'} + \varepsilon_p)^\alpha}, \tag{9}$$

where newly generated data have priority $p_i = p_{\max}$ (Schaul et al., 2015), to ensure that each is used for at least one update. $P(i)$ is the replay probability with floor $\varepsilon_p > 0$ and exponent α. Positive samples ($\widetilde{r}_i > 0$) are replayed first when π_θ fits them worse than π_{ref}, e.g., after interference from other updates. We denote by η the resulting joint replay law of rollouts (c, x_0), which reweights the loss without changing the minimizer (Theorem 1).

4.4 Why VIRM Works

Let $P_{\bar{\pi}_{\mathrm{ema}}}$ be the joint law of $c \sim \mathcal{D}$ followed by $x_0 \sim \bar{\pi}_{\mathrm{ema}}(\cdot \mid c)$. The first result concerns the uncentered objective; by Eq. (6), centering leaves its root unchanged.

Theorem 1 (Identification under replay)**.** *Fix the anchor and targets. If η covers $P_{\bar{\pi}_{\mathrm{ema}}}$, candidate policies are supported by $\bar{\pi}_{\mathrm{ema}}$, and $(\pi^\star, \tau \log Z)$ is realizable, then every zero-loss solution of Eq. (5) satisfies*

$$\pi_\theta(\cdot \mid c) = \pi^\star(\cdot \mid c), \qquad z(c) = \tau \log Z(c) \quad \textit{for } \mathcal{D}\textit{-a.e. } c. \tag{10}$$

The proof is given in Appendix A.3. Uniform, prioritized, and repeated replay thus identify the same Gibbs policy whenever they cover the anchor, which the floor ε_p ensures in the population limit; if prompts are drawn independently from $\mathcal{D}$, it suffices to cover x_0 for each prompt. If the model class cannot represent the target, prioritization changes which approximation is reached, and importance weighting is needed to fit a specific distribution.

For the practical objective, let ν be the training law of Eq. (8). Let $\varepsilon_{\mathrm{ELBO}}$, ε_z, and ε_r be the $L_2(\nu)$ errors of the element-averaged ELBO ratio $\tau\bar{q}_t$ (including its single-draw error and the $w(t) = 1$ surrogate gap), of the averaged head $\bar{z}_\phi$ relative to $z^\star$, and of the centered reward estimate $\widetilde{r}_i$ relative to the exact centered reward $r - b(c)$, and define $\Delta = \sqrt{\mathcal{L}_{\mathrm{VIRM}}} + \varepsilon_{\mathrm{ELBO}} + \varepsilon_z + \varepsilon_r$.

Theorem 2 (Approximate recovery)**.** *Let P_θ be the joint law induced by $\mathcal{D}$ and π_θ. If P_θ is covered by η with density ratio at most C_η, the candidate policy is supported by $\bar{\pi}_{\mathrm{ema}}$, and $\Delta < \infty$, then*

$$\mathbb{E}_{c\sim\mathcal{D}} D_{\mathrm{KL}}(\pi_\theta \| \pi^\star) \le \frac{\sqrt{C_\eta}}{\tau}\Delta. \tag{11}$$

The proof is given in Appendix A.4. The divergence is thus controlled by the fitting residual and one error per approximation of Section 4.2, and clipping and the floor raise C_η by a factor of at most $((p_{\max} + \varepsilon_p)/\varepsilon_p)^\alpha$ relative to uniform replay. The same appendix bounds the regularized objective gap by $\sqrt{C_\eta}\Delta$ and derives total-variation and bounded-reward consequences.

5 Experiments

Setup. We employ SD3.5-Medium (Esser et al., 2024) and FLUX.2-klein-base-9B (FLUX.2-9B) (Labs, 2025) as base models to cover diverse architectures and scales. We compare our method with competitive baselines: Flow-GRPO (Liu et al., 2025a), GRPO-Guard (Wang et al., 2026), DiffusionNFT (Zheng et al., 2026), AWM (Xue et al., 2026), and the recently proposed Flow-DPPO (Ping et al., 2026). For Flow-DPPO, we report its CPS variant (Flow-DPPO + CPS), which is stronger on most metrics in the original paper, and refer to it as Flow-DPPO for simplicity. Flow-GRPO and GRPO-Guard apply GRPO to flow models through ODE-to-SDE sampling, whereas DiffusionNFT and AWM directly optimize the velocity field through the forward process.

Table 1: Performance comparison on SD3.5 and FLUX2-9B. The training is applied on the In-Domain (GenEval2). The Out-of-Domain (PickScore) prompts are only used for evaluation. † indicates the results drawn from Flow-DPPO (Ping et al., 2026).

	In-Domain (GenEval2)				Out-of-Domain (PickScore)		
Model	GenEval2	CLIP	PickScore	HPSv2	CLIP	PickScore	HPSv2
Pretrained baselines (before RL)							
SD3.5-medium†	12.4	0.250	21.00	0.213	0.244	19.99	0.210
FLUX2-klein-base-9B†	25.4	0.281	20.92	0.228	0.254	20.05	0.230
SD3.5-medium, multi-reward RL fine-tuning							
Flow-GRPO†	39.9	0.358	25.09	0.399	0.273	22.07	0.349
GRPO-Guard†	47.8	0.353	25.64	0.409	0.272	22.32	0.354
Diffusion-NFT†	42.5	0.334	25.30	0.394	0.269	22.52	0.355
AWM	43.1	0.349	25.53	0.401	0.272	22.47	0.352
Flow-DPPO†	51.6	0.369	25.72	0.415	0.279	22.51	0.361
VIRM (Ours)	**53.2**	**0.372**	**25.90**	**0.421**	**0.283**	**22.57**	**0.366**
FLUX2-klein-base-9B, multi-reward RL fine-tuning							
Flow-GRPO†	46.8	0.371	25.61	0.412	0.277	22.62	0.357
GRPO-Guard†	49.0	0.375	25.27	0.411	0.269	21.99	0.349
Diffusion-NFT†	47.3	0.336	24.87	0.389	0.274	22.47	0.351
AWM	48.8	0.369	25.12	0.402	0.278	22.53	0.355
Flow-DPPO†	55.2	0.386	**26.15**	**0.427**	0.287	22.97	0.370
VIRM (Ours)	**65.4**	**0.389**	26.10	0.423	**0.291**	**23.10**	**0.374**

Training and Evaluation Protocol. Following the setup of Flow-DPPO (Ping et al., 2026), we train on 20k synthetic GenEval2 prompts and use PickScore (Kirstain et al., 2023), CLIP score (Radford et al., 2021), HPSv2 (Wu et al., 2023), and Soft TIFA_{GM} score on GenEval2 as rewards for multi-reward optimization. CLIP score measures text–image alignment, PickScore and HPSv2 are learned models of human preference, and GenEval2 (Kamath et al., 2025) measures compositional prompt following, such as object counts and attribute binding. We evaluate the trained models on the 800 official GenEval2 prompts (in-domain) and on PickScore prompts (out-of-domain). The evaluation prompts are never used for training; they test whether gains transfer beyond the training distribution. Since the methods differ in how many updates they take per rollout, we compare training cost in GPU hours rather than in steps. Implementation details and hyperparameters are given in Appendix C.

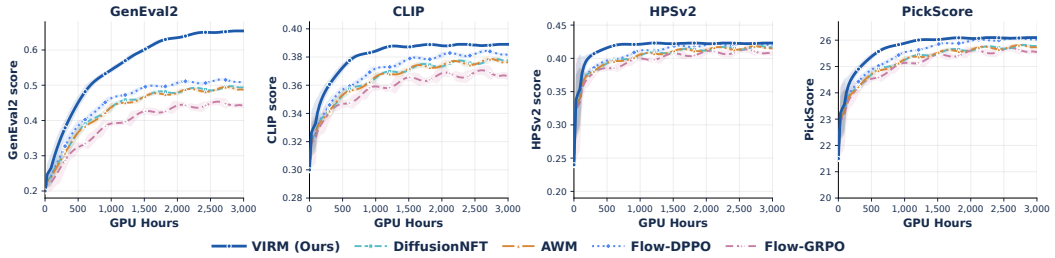


Figure 4: Learning curves on FLUX.2-9B under multi-reward optimization. VIRM achieves the highest training efficiency while attaining the best final score.

5.1 Results Analysis

VIRM achieves the best compositional alignment on both backbones. Table 1 shows that VIRM attains the highest GenEval2 score on both models. On SD3.5-Medium it improves the base model from 12.4 to 53.2, outperforming all the competitive baselines. The margin widens on the larger FLUX.2-9B, where VIRM raises GenEval2 from 25.4 to 65.4, beating the strongest baseline, Flow-DPPO, by **10.2** absolute points.

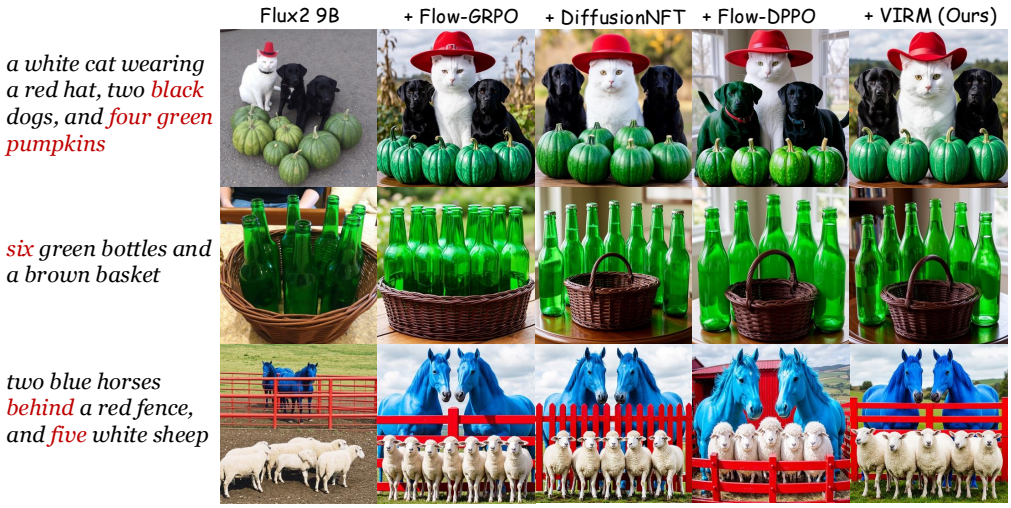


Figure 5: Qualitative comparison on FLUX.2-9B for compositional prompts. Words in red mark counts and attributes that the samples must satisfy.

VIRM generalizes well across different metrics. The results in Table 1 show that all RL methods substantially improve the downstream performance of the base models. VIRM gives the strongest overall results, and the advantage is consistent across architectures: VIRM is best on every metric for SD3.5-Medium, and on FLUX.2-9B it leads on most metrics and achieves superior average scores. The large GenEval2 gain on FLUX.2-9B therefore does not come at the cost of other rewards.

Improvements transfer to out-of-domain prompts. On the held-out PickScore prompts, VIRM demonstrates the best generalization performance across different metrics and backbones. The gains of VIRM are thus not confined to the training distribution or attributable to in-domain reward hacking.

VIRM is substantially more sample-efficient and remains stable. As shown in Fig. 4, VIRM rises quickly and then plateaus without degradation. The advantage is most pronounced on GenEval2: VIRM surpasses the final score of every baseline within fewer than 1,000 GPU hours, i.e., with less than $\mathbf{1/3}$ of the training budget. VIRM also achieves a superior final score, outperforming the strongest baseline, Flow-DPPO, by around $\mathbf{40\%}$ in terms of relative improvements over the base model. On the other metrics, VIRM converges consistently faster, whereas the baselines reach comparable values only near the end of training. Prioritized replay (Section 4.3) is a key contributor to this efficiency: removing the buffer or replaying uniformly yields lower scores (Fig. 7(b)).

5.1.1 Qualitative Results

VIRM demonstrates better instruction-following ability. Fig. 5 compares FLUX.2-9B samples before and after RL fine-tuning on GenEval2-style prompts that combine counts, colors, and spatial relations. The pretrained model and the baselines often miscount objects, e.g., generating more bottles or sheep than requested, or bind an attribute to the wrong object, e.g., Flow-DPPO renders one of the black dogs green. VIRM follows the specified counts and attributes more closely, in line with its GenEval2 gains in Table 1.

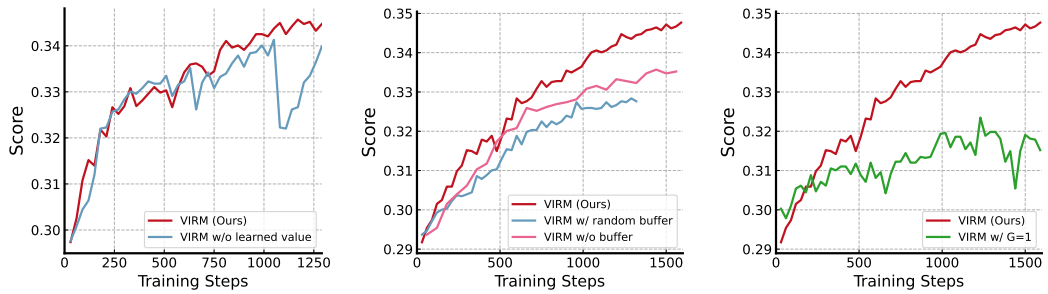


Figure 6: Sample diversity on FLUX.2-9B for simple prompts: Flow-DPPO (top) and VIRM (bottom).

VIRM preserves sample diversity where Flow-DPPO collapses. Fig. 6 shows four samples each for the prompts "A dog" and "A chair". Flow-DPPO collapses to near-identical close-ups of the same golden dog and the same wooden chair, whereas VIRM varies breed, pose, framing, and chair design while remaining faithful to the prompt. This is consistent with the KL-regularized target of VIRM, which tilts the anchor toward high reward rather than concentrating on a single mode (Section 4.1).

5.2 Ablation Studies

We ablate the three stabilizing components of Sections 4.2 and 4.3: the token-wise value head, reducible-loss prioritized replay, and group-wise reward centering.



(a) Removing the learned value z_ϕ. (b) Replay variants. (c) Removing reward centering.

Figure 7: Ablations of VIRM. (a): "w/o learned value" remove z_ϕ in the final loss in Eq. (8). (b): "w/o buffer" trains without replay, "w/ random buffer" replays uniformly. (c): "w/ G=1" trains without centering (raw rewards).

The learned value head is necessary for stable training. Fig. 7(a) compares VIRM with a variant that removes the value head, i.e., fixes $z_\phi \equiv 0$. Both curves track each other early on, but the variant without the head becomes unstable after roughly 1,000 steps, collapses sharply, and only partially recovers, whereas VIRM improves steadily. This is consistent with the role of the token-wise value head z_ϕ (Section 4.2): the normalizer is necessary for the regression target of Eq. (8) to be valid. Without a learned z_ϕ, the resulting error $\varepsilon_z = \|z^\star\|_{L_2(\nu)}$ enters the bound of Theorem 2. Training cannot reduce this error, and repeated replay updates may reinforce the resulting systematic bias.

Reducible-loss prioritized replay outperforms both no replay and uniform replay. As shown in Fig. 7(b), reuse alone does not help: replaying uniformly from the same buffer falls even below training without replay. The gain of the buffer comes from prioritized reuse. We attribute this to uniform replay spending updates on samples that are already fit or dominated by reward and ELBO noise, whereas the signed reducible-loss priority of Eq. (9) concentrates reuse on outputs that the policy can still fit better.

Group-wise reward centering is essential. Training without centering (G=1, raw rewards) yields the lowest and most volatile result in Fig. 7(c), with sharp drops and little net improvement. With raw rewards, the between-prompt offset $b(c)$ dominates the regression target and leaks into the ratio term; centering removes this offset without changing the optimal target $\pi^\star$ (Eq. (6)).

6 Conclusion

In this paper, we presented VIRM, which casts reward fine-tuning of diffusion and flow models as regression of the policy-implied value onto external reward feedback. Group-wise reward centering, an ELBO-based ratio estimate, a token-wise value head, and reducible-loss prioritized replay together enable high-UTD replay without importance weights or stored trajectories. On GenEval2, VIRM raises FLUX.2-9B from 25.4 to 65.4, 10.2 points above the strongest baseline, and it attains the best out-of-domain scores across different base models.

Limitations. The replay buffer has finite capacity, and the EMA anchor must be updated or reset periodically so that the replayed data keep covering the current policy, as our guarantees require. Moreover, our experiments are limited to text-to-image generation. Validating the superior effects of VIRM on video generation and robotics tasks is left for future work.

AI USE STATEMENT

In this work, we used LLM-based assistants to help with writing, including polishing and condensing text, proofreading, checking the consistency of notation and derivations, and LaTeX formatting. The research ideas, method, experiments, and results are the authors' own. The authors reviewed and verified all AI-assisted text and derivations and take full responsibility for the content of this work.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Brian R. Bartoldson, Siddarth Venkatraman, James Diffenderfer, Moksh Jain, Tal Ben-Nun, Seanie Lee, Minsu Kim, Johan Obando-Ceron, Yoshua Bengio, and Bhavya Kailkhura. Trajectory balance with asynchrony: Decoupling exploration and learning for fast, scalable LLM post-training. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=VwPt1WDQNB.

Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J Hu, Mo Tiwari, and Emmanuel Bengio. Gflownet foundations. *Journal of Machine Learning Research*, 24(210):1–55, 2023.

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.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. URL https://openai.com/research/video-generation-models-as-world-simulators.

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. In *Forty-third International Conference on Machine Learning*, 2026.

Haoyou Deng, Keyu Yan, Chaojie Mao, Xiang Wang, Yu Liu, Changxin Gao, and Nong Sang. Densegrpo: From sparse to dense reward for flow matching model alignment. *arXiv preprint arXiv:2601.20218*, 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.

Haoran He, Emmanuel Bengio, Qingpeng Cai, and Ling Pan. Random policy evaluation uncovers policies of generative flow networks. In *Proceedings of the 42nd International Conference on Machine Learning*, Proceedings of Machine Learning Research, pp. 22351–22367. PMLR, 13–19 Jul 2025a.

Haoran He, Jiajun Liang, Xintao Wang, Pengfei Wan, Di Zhang, Kun Gai, and Ling Pan. Scaling image and video generation via test-time evolutionary search. *arXiv preprint arXiv:2505.17618*, 2025b.

Haoran He, Yuxiao Ye, Jie Liu, Jiajun Liang, Zhiyong Wang, Ziyang Yuan, Xintao Wang, Hangyu Mao, Pengfei Wan, and Ling Pan. Gardo: Reinforcing diffusion models without reward hacking. *arXiv preprint arXiv:2512.24138*, 2025c.

Xiaoxuan He, Siming Fu, Yuke Zhao, Wanli Li, Jian Yang, Dacheng Yin, Fengyun Rao, and Bo Zhang. Tempflow-grpo: When timing matters for grpo in flow models. *arXiv preprint arXiv:2508.04324*, 2025d.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

Amita Kamath, Kai-Wei Chang, Ranjay Krishna, Luke Zettlemoyer, Yushi Hu, and Marjan Ghazvininejad. Geneval 2: Addressing benchmark drift in text-to-image evaluation. *arXiv preprint arXiv:2512.16853*, 2025.

Jaihoon Kim, Taehoon Yoon, Jisung Hwang, and Minhyuk Sung. Inference-time scaling for flow models via stochastic generation and rollover budget forcing. *arXiv preprint arXiv:2503.19385*, 2025.

Diederik Kingma and Ruiqi Gao. Understanding Diffusion Objectives as the ELBO with Simple Data Augmentation. In *Neural Information Processing Systems*, 2023. URL https://mlanthology.org/neurips/2023/kingma2023neurips-understanding/.

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.

Branislav Kveton, Anup Rao, Subhojyoti Mukherjee, Krishna Kumar Singh, and Viet Dac Lai. Advantageflow: Advantage-weighted least squares for rl in flow models. *arXiv preprint arXiv:2605.26013*, 2026.

Black Forest Labs. FLUX.2: Frontier Visual Intelligence. https://bfl.ai/blog/flux-2, 2025.

Jeongjae Lee and Jong Chul Ye. Pcpo: Proportionate credit policy optimization for aligning image generation models. *arXiv preprint arXiv:2509.25774*, 2025.

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.

Junzhe Li, Yutao Cui, Tao Huang, Yinping Ma, Chun Fan, Miles Yang, and Zhao Zhong. Mixgrpo: Unlocking flow-based grpo efficiency with mixed ode-sde. *arXiv preprint arXiv:2507.21802*, 2025a.

Yuming Li, Yikai Wang, Yuying Zhu, Zhongyu Zhao, Ming Lu, Qi She, and Shanghang Zhang. Branchgrpo: Stable and efficient grpo with structured branching in diffusion models. *arXiv preprint arXiv:2509.06040*, 2025b.

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

Jiashun Liu, Johan Obando-Ceron, Han Lu, Yancheng He, Weixun Wang, Wenbo Su, Bo Zheng, Pablo Samuel Castro, Aaron Courville, and Ling Pan. Asymmetric proximal policy optimization: mini-critics boost LLM reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=0vgzrcv4Dr.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di ZHANG, and Wanli Ouyang. Flow-GRPO: Training flow matching models via online RL. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025a. URL https://openreview.net/forum?id=oCBKGw5HNf.

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

Zhen Liu, Tim Z. Xiao, Weiyang Liu, Carles Domingo-Enrich, and Dinghuai Zhang. Value gradient guidance for flow matching alignment. In *NeurIPS*, 2025b.

Yihong Luo, Tianyang Hu, and Jing Tang. Reinforcing diffusion models by direct group preference optimization. In *International Conference on Learning Representations*, volume 2026, pp. 21438–21454, 2026.

Martin Marek and Max Ryabinin. Score centering stabilizes off-policy reinforcement learning. *arXiv preprint arXiv:2609.20807*, 2026.

Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. Bridging the gap between value and policy based reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

Abhishek Naik, Yi Wan, Manan Tomar, and Richard S. Sutton. Reward centering. In *Reinforcement Learning Conference*, 2024. URL https://openreview.net/forum?id=AVAcUmAhXI.

Oren Peer, Chen Tessler, Nadav Merlis, and Ron Meir. Ensemble bootstrapping for q-learning. In *International conference on machine learning*, pp. 8454–8463. PMLR, 2021.

Bo-Wen Ping, Xiangxin Zhou, Penghui Qi, Minnan Luo, Liefeng Bo, and Tianyu Pang. Flow-dppo: Divergence proximal policy optimization for flow matching models. *ArXiv*, abs/2606.11025, 2026. URL https://api.semanticscholar.org/CorpusID:289132796.

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.

Yash Savani, Branislav Kveton, Yuchen Liu, Yilin Wang, Jing Shi, Subhojyoti Mukherjee, Nikos Vlassis, and Krishna Kumar Singh. Stepwise credit assignment for grpo on flow-matching models. *arXiv preprint arXiv:2603.28718*, 2026.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv: Learning*, 2015. URL https://api.semanticscholar.org/CorpusID:13022595.

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

Team Seedance, De Chen, Liyang Chen, Xin Chen, Ying Chen, Zhuo Chen, Zhuowei Chen, Feng Cheng, Tianheng Cheng, Yufeng Cheng, et al. Seedance 2.0: Advancing video generation for world complexity. *arXiv preprint arXiv:2604.14148*, 2026.

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.

Saurabh Singh and Ian Fischer. Stochastic sampling from deterministic flow models. *arXiv preprint arXiv:2410.02217*, 2024.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Shivakanth Sujit, Somjit Nath, Pedro Braga, and Samira Ebrahimi Kahou. Prioritizing samples in reinforcement learning with reducible loss. *Advances in Neural Information Processing Systems*, 36:23237–23258, 2023.

Masatoshi Uehara, Yulai Zhao, Tommaso Biancalani, and Sergey Levine. Understanding reinforcement learning-based fine-tuning of diffusion models: A tutorial and review. *arXiv preprint arXiv:2407.13734*, 2024.

Siddarth Venkatraman, Moksh Jain, Luca Scimeca, Minsu Kim, Marcin Sendera, Mohsin Hasan, Luke Rowe, Sarthak Mittal, Pablo Lemos, Emmanuel Bengio, et al. Amortizing intractable inference in diffusion models for vision, language, and control. *Advances in neural information processing systems*, 37:76080–76114, 2024.

Feng Wang and Zihao Yu. Coefficients-preserving sampling for reinforcement learning with flow matching. *arXiv preprint arXiv:2509.05952*, 2025.

Jing Wang, Jiajun Liang, Jie Liu, Henglin Liu, Gongye Liu, Jun Zheng, Wanyuan Pang, Ao Ma, Zhenyu Xie, Xintao Wang, Meng Wang, Pengfei Wan, and Xiaodan Liang. Grpo-guard: Mitigating implicit over-optimization in flow matching via regulated clipping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5988–5998, June 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.

Shuchen Xue, Chongjian GE, Shilong Zhang, Yichen Li, and Zhi-Ming Ma. Advantage weighted matching: Aligning RL with pretraining in diffusion models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=nLY2pOYBrJ.

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*, 2025.

Liyu Zhang, Kehan Li, Tingrui Han, Tao Zhao, Yuxuan Sheng, Shibo He, and Chao Li. Op-grpo: Efficient off-policy grpo for flow-matching models. *arXiv preprint arXiv:2604.04142*, 2026.

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. DiffusionNFT: Online diffusion reinforcement with forward process. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=VJZ477R89F.

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

A Theoretical Details

Throughout, the anchor $\bar{\pi}_{\text{ema}}$ is fixed, $Z(c) = \mathbb{E}_{x_0 \sim \bar{\pi}_{\text{ema}}(\cdot|c)}[e^{r(x_0,c)/\tau}]$ is the partition function of Eq. (2), assumed to satisfy $0 < Z(c) < \infty$, and $z^\star(c) = \tau \log Z(c) - b(c)$ is the centered normalizer of Eq. (6).

A.1 Uncentered Gibbs Solution

Fix c and omit it from the notation. For every $\pi \ll \bar{\pi}_{\text{ema}}$, substituting $\log \pi^\star = \log \bar{\pi}_{\text{ema}} + r/\tau - \log Z$ from Eq. (2) gives

$$\tau D_{\text{KL}}(\pi \| \pi^\star) = \tau D_{\text{KL}}(\pi \| \bar{\pi}_{\text{ema}}) - \mathbb{E}_\pi[r] + \tau \log Z = \tau \log Z - \mathcal{J}_c(\pi). \tag{12}$$

Since the divergence vanishes only at $\pi = \pi^\star$, $\pi^\star$ is the unique maximizer in Eq. (2), with $\mathcal{J}_c(\pi^\star) = \tau \log Z$ and

$$\mathcal{J}_c(\pi^\star) - \mathcal{J}_c(\pi) = \tau D_{\text{KL}}(\pi \| \pi^\star). \tag{13}$$

Taking logarithms of $\pi^\star$ yields Eq. (4).

A.2 Centering Invariance

For any shift $s(c)$ that does not depend on x_0, $\mathbb{E}_\pi[r - s(c)] - \tau D_{\text{KL}}(\pi \| \bar{\pi}_{\text{ema}}) = \mathcal{J}_c(\pi) - s(c)$, so the maximizer $\pi^\star$ is unchanged, and the normalizer of the shifted problem is

$$\tau \log \mathbb{E}_{x_0 \sim \bar{\pi}_{\text{ema}}(\cdot|c)} \left[e^{(r(x_0,c) - s(c))/\tau} \right] = \tau \log Z(c) - s(c). \tag{14}$$

For the population baseline $s(c) = b(c) = \mathbb{E}_{\bar{\pi}_{\text{ema}}}[r \mid c]$, subtracting $b(c)$ from both sides of Eq. (4) gives the centered identity in Eq. (6), the right-hand side above equals $z^\star(c)$, and three properties follow. First, by the law of total variance under $P_{\bar{\pi}_{\text{ema}}}$,

$$\text{Var}(r) = \mathbb{E}_c[\text{Var}(r \mid c)] + \text{Var}_c(b(c)), \qquad \text{Var}(r - b(c)) = \mathbb{E}_c[\text{Var}(r \mid c)], \tag{15}$$

so centering removes the between-prompt variance. This identity is descriptive; it does not imply that empirical group centering reduces every component of the gradient variance. Second, since $\mathbb{E}_{\bar{\pi}_{\text{ema}}}[r - b(c) \mid c] = 0$, Jensen's inequality gives $z^\star(c) \geq 0$. Third, expanding $e^{(r-b(c))/\tau}$ and $\log(1+x)$ to second order in $(r-b(c))/\tau$ gives $z^\star(c) \approx \text{Var}_{\bar{\pi}_{\text{ema}}}(r \mid c)/(2\tau)$ when the within-prompt variation of r is small relative to τ.

A.3 Proof of Theorem 1

Proof. Zero loss in Eq. (5) means that

$$z(c) + \tau \log \frac{\pi_\theta(x_0 \mid c)}{\bar{\pi}_{\text{ema}}(x_0 \mid c)} = r(x_0, c) \tag{16}$$

holds η-almost surely, and hence $P_{\bar{\pi}_{\text{ema}}}$-almost surely because $P_{\bar{\pi}_{\text{ema}}} \ll \eta$. Thus, for $\mathcal{D}$-almost every c, $\pi_\theta(x_0 \mid c) = \bar{\pi}_{\text{ema}}(x_0 \mid c)\, e^{(r(x_0,c) - z(c))/\tau}$ for $\bar{\pi}_{\text{ema}}(\cdot \mid c)$-almost every x_0. Since $\pi_\theta(\cdot \mid c) \ll \bar{\pi}_{\text{ema}}(\cdot \mid c)$, the exceptional $\bar{\pi}_{\text{ema}}$-null set is also π_θ-null, and integrating over x_0 gives

$$1 = e^{-z(c)/\tau}\, \mathbb{E}_{x_0 \sim \bar{\pi}_{\text{ema}}(\cdot|c)} \left[e^{r(x_0,c)/\tau} \right] = e^{(\tau \log Z(c) - z(c))/\tau}. \tag{17}$$

Hence $z(c) = \tau \log Z(c)$, and substituting back gives $\pi_\theta(\cdot \mid c) = \pi^\star(\cdot \mid c)$ by Eq. (2). Realizability ensures that such a zero-loss solution exists. By Appendix A.2, replacing $(r, \tau \log Z)$ with $(r - b(c), z^\star)$ leaves this policy root unchanged. □

A.4 Proof of Theorem 2

Proof. Let $\bar{\delta} = d^{-1} \mathbf{1}^\top \boldsymbol{\delta} = \bar{z}_\phi + \tau \bar{q}_t - \widetilde{r}$ be the element mean of the residual in Eq. (8), where $\bar{z}_\phi = d^{-1} \mathbf{1}^\top z_\phi$ is used only in the analysis. Because

$$\tfrac{1}{d} \|\boldsymbol{\delta}\|_2^2 = \bar{\delta}^2 + \tfrac{1}{d} \|\boldsymbol{\delta} - \bar{\delta} \mathbf{1}\|_2^2 \geq \bar{\delta}^2, \tag{18}$$

we have $\|\bar{\delta}\|_{L_2(\nu)} \le \sqrt{\mathcal{L}_{\mathrm{VIRM}}}$. The approximation errors in Theorem 2 are

$$\varepsilon_{\mathrm{ELBO}} = \Big\| \tau \bar{q}_t - \tau \log \frac{\pi_\theta}{\bar{\pi}_{\mathrm{ema}}} \Big\|_{L_2(\nu)}, \quad \varepsilon_z = \|\bar{z}_\phi - z^\star\|_{L_2(\nu)}, \quad \varepsilon_r = \|\widetilde{r} - (r - b(c))\|_{L_2(\nu)}. \tag{19}$$

Thus ε_r is the error of the group-centered estimate $\widetilde{r}$ relative to the exact centered reward $r - b(c)$, i.e., of the group mean relative to $b(c)$. Because ν includes the draw (t, ϵ), $\varepsilon_{\mathrm{ELBO}}$ contains both the deviation of the single draw $\tau \bar{q}_t$ from its (t, ϵ)-average and the gap between this average and the log-ratio. Writing $\tau \log(\pi^\star / \bar{\pi}_{\mathrm{ema}}) = r - b(c) - z^\star$ by Eq. (6), we obtain

$$\tau \log \frac{\pi_\theta}{\pi^\star} = \Big(\tau \log \frac{\pi_\theta}{\bar{\pi}_{\mathrm{ema}}} - \tau \bar{q}_t \Big) + \bar{\delta} + \big(z^\star - \bar{z}_\phi \big) + \big(\widetilde{r} - (r - b(c)) \big), \tag{20}$$

and the triangle inequality gives

$$\Big\| \tau \log \frac{\pi_\theta}{\pi^\star} \Big\|_{L_2(\nu)} \le \Delta. \tag{21}$$

The log-ratio is well defined because π_θ is supported by $\bar{\pi}_{\mathrm{ema}}$, which has the same support as $\pi^\star$; it depends only on (c, x_0), whose law under ν is η, so its $L_2(\nu)$ and $L_2(\eta)$ norms coincide. Let $h = dP_\theta / d\eta$. Since $h \le C_\eta$ and $\mathbb{E}_\eta[h] = 1$, we have $\mathbb{E}_\eta[h^2] \le C_\eta$, and the Cauchy–Schwarz inequality gives

$$\tau \, \mathbb{E}_c D_{\mathrm{KL}}(\pi_\theta \| \pi^\star) = \mathbb{E}_\eta \Big[h \, \tau \log \frac{\pi_\theta}{\pi^\star} \Big] \le \sqrt{\mathbb{E}_\eta[h^2]} \, \Big\| \tau \log \frac{\pi_\theta}{\pi^\star} \Big\|_{L_2(\eta)} \le \sqrt{C_\eta} \, \Delta. \tag{22}$$

Dividing by τ proves Eq. (11). By Eq. (13), the regularized objective gap satisfies $\mathbb{E}_c[\mathcal{J}_c(\pi^\star) - \mathcal{J}_c(\pi_\theta)] \le \sqrt{C_\eta} \, \Delta$. Pinsker's and Jensen's inequalities further give

$$\mathbb{E}_c \big[\mathrm{TV}(\pi_\theta, \pi^\star) \big] \le \sqrt{\frac{\sqrt{C_\eta} \, \Delta}{2\tau}}, \tag{23}$$

and, if $|r| \le r_{\max}$, the expected rewards of π_θ and $\pi^\star$ differ by at most $2 r_{\max}$ times this bound. □

Effect of prioritization on C_η. In the population limit, uniform replay of anchor rollouts has law $P_{\bar{\pi}_{\mathrm{ema}}}$, and the replay probability $P(i)$ of Eq. (9) reweights it by

$$\frac{d\eta}{dP_{\bar{\pi}_{\mathrm{ema}}}} = \frac{(p + \varepsilon_p)^\alpha}{\mathbb{E}_{P_{\bar{\pi}_{\mathrm{ema}}}}[(p + \varepsilon_p)^\alpha]} \ge \Big(\frac{\varepsilon_p}{p_{\max} + \varepsilon_p} \Big)^\alpha > 0, \tag{24}$$

where $p \in [0, p_{\max}]$ is the priority of (c, x_0). Hence η covers $P_{\bar{\pi}_{\mathrm{ema}}}$ as required by Theorem 1, and if $dP_\theta / dP_{\bar{\pi}_{\mathrm{ema}}} \le C_{\mathrm{unif}}$, then

$$\frac{dP_\theta}{d\eta} = \frac{dP_\theta}{dP_{\bar{\pi}_{\mathrm{ema}}}} \, \frac{dP_{\bar{\pi}_{\mathrm{ema}}}}{d\eta} \le \Big(\frac{p_{\max} + \varepsilon_p}{\varepsilon_p} \Big)^\alpha C_{\mathrm{unif}}. \tag{25}$$

Prioritization thus inflates the coverage constant of uniform replay by at most this factor.

B Training Algorithm

Algorithm 1 summarizes VIRM; all quantities are defined in Section 4. Each training step evaluates v_θ, the anchor $v_{\bar{\theta}_{\mathrm{ema}}}$, and the frozen v_{ref} on the same noisy state, and back-propagates only through v_θ and the value head. In code, τq_t is the raw element-wise squared-error difference multiplied by $-\tau d \, w(t)$, and the loss is averaged over elements.

Algorithm 1 VIRM training

Require: pretrained parameters θ_{ref}, prompt law $\mathcal{D}$, group size G, temperature τ, number of heads H, priority parameters $(\alpha, \varepsilon_p, p_{\max})$, interval length U (training steps)

$\theta \leftarrow \theta_{\mathrm{ref}}$, $\bar{\theta}_{\mathrm{ema}} \leftarrow \theta_{\mathrm{ref}}$; initialize the value heads ϕ
for each interval **do**
  Sample prompts $c \sim \mathcal{D}$ and G outputs per prompt from the anchor $\bar{\pi}_{\mathrm{ema}}(\cdot \mid c)$; score their rewards
  Center the rewards by their group means (Eq. (6)) and store $(c_i, x_{0_i}, \widetilde{r}_i)$ in an empty buffer $\mathcal{B}$ with priorities $p_i = p_{\max}$
  for each of the U training steps **do**
    Draw a minibatch $i \sim P(i)$ (Eq. (9)), and for each i one $t \sim \mathcal{U}[0,1]$ and $\epsilon \sim \mathcal{N}(0, I)$
    Set $x_t = (1-t)x_{0_i} + t\epsilon$ and $u^\star = \epsilon - x_{0_i}$; compute $q_t \in \mathbb{R}^d$ with v_θ and $v_{\bar{\theta}_{\mathrm{ema}}}$ on x_t (Eq. (7))
    Form z_ϕ with the straight-through ensemble rule over a randomly sampled nonempty subset $S \subseteq [H]$
    Refresh p_i of the minibatch from $\mathrm{sign}(\widetilde{r}_i)$, $\mathcal{L}_{\theta,i}$, and $\mathcal{L}_{\mathrm{ref},i}$ on the same draw (Eq. (9))
    Update (θ, ϕ) by a gradient step on $\mathcal{L}_{\mathrm{VIRM}}$ (Eq. (8))
  end for
  Update $\bar{\theta}_{\mathrm{ema}}$ by one EMA step toward θ (Appendix C) and clear $\mathcal{B}$
end for

B.1 Design Notes

Single-draw loss. Fix an output and a head $z\mathbf{1}$ that is constant across elements, and let $a = z - \widetilde{r}$. Applying Eq. (18) to $\boldsymbol{\delta} = a\mathbf{1} + \tau q_t$ and averaging over (t, ϵ) gives

$$\mathbb{E}_{t,\epsilon}\Big[\tfrac{1}{d}\|a\mathbf{1} + \tau q_t\|_2^2\Big] = \big(a + \tau\,\mathbb{E}_{t,\epsilon}[\bar{q}_t]\big)^2 + \tau^2\,\mathrm{Var}_{t,\epsilon}(\bar{q}_t) + \tau^2\,\mathbb{E}_{t,\epsilon}\Big[\tfrac{1}{d}\|q_t - \bar{q}_t\mathbf{1}\|_2^2\Big]. \tag{26}$$

Relative to the loss of the averaged ratio (first term), a single draw thus adds the variance of $\tau\bar{q}_t$ and the element-wise dispersion of τq_t. By Eq. (7), $q_{t,k} = -d\,w(t)(v_\theta - v_{\bar{\theta}_{\mathrm{ema}}})_k(v_\theta + v_{\bar{\theta}_{\mathrm{ema}}} - 2u^\star)_k$ and $\bar{q}_t = -w(t)(v_\theta - v_{\bar{\theta}_{\mathrm{ema}}})^\top(v_\theta + v_{\bar{\theta}_{\mathrm{ema}}} - 2u^\star)$, so by the Cauchy–Schwarz inequality

$$\mathrm{Var}_{t,\epsilon}(\bar{q}_t) \le \mathbb{E}_{t,\epsilon}[\bar{q}_t^2] \le \mathbb{E}_{t,\epsilon}\big[w(t)^2\|v_\theta - v_{\bar{\theta}_{\mathrm{ema}}}\|_2^2\,\|v_\theta + v_{\bar{\theta}_{\mathrm{ema}}} - 2u^\star\|_2^2\big]. \tag{27}$$

Because both models are evaluated on the same $(x_t, u^\star)$, both extra terms vanish where $v_\theta = v_{\bar{\theta}_{\mathrm{ema}}}$, so they favor anchor-proximal policies. A head that reads (x_t, t) can absorb part of the t-dependent component of q_t at the price of deviating from $z^\star$, which ε_z records; Theorem 2 accounts for the remainder through $\sqrt{\mathcal{L}_{\mathrm{VIRM}}}$ and $\varepsilon_{\mathrm{ELBO}}$.

Element-wise advantages. The head reads detached features, so z_ϕ passes no gradient to θ. Since $\nabla_\theta q_{t,k} = -d\,w(t)\nabla_\theta(v_\theta - u^\star)_k^2$ by Eq. (7),

$$\nabla_\theta\,\tfrac{1}{d}\|\boldsymbol{\delta}_{i,t}\|_2^2 = 2\tau\,w(t)\sum_{k=1}^{d} A_{i,t,k}\,\nabla_\theta\big(v_\theta(x_t, t, c_i) - u^\star\big)_k^2, \qquad A_{i,t,k} = \widetilde{r}_i - z_{\phi,k} - \tau q_{t,k}. \tag{28}$$

Each step is thus a flow-matching update on x_{0_i} in which element k carries its own advantage $A_{i,t,k}$: a gradient step lowers the error of element k when the observed reward exceeds the reward implied at that element, raises it otherwise, and stops once the two agree. The token-wise head thus sets a separate reward target $\widetilde{r}_i - z_{\phi,k}$ for each element, whereas a head that outputs only the average $\bar{z}_\phi$ gives every element the same target $\widetilde{r}_i - \bar{z}_\phi$. Regressing only the element mean $\bar{\delta}_{i,t}$ would further give every element the common advantage $-\bar{\delta}_{i,t} = \widetilde{r}_i - \bar{z}_\phi - \tau\bar{q}_t$, i.e., a uniformly weighted flow-matching update.

$\bar{\pi}_{\mathrm{ema}}$ update schedule. Within an interval the buffer is fixed while π_θ moves, so the density-ratio bound C_η in Theorem 2 grows with the drift. The target itself stays covered: if $|r| \le r_{\max}$, then $\pi^\star/\bar{\pi}_{\mathrm{ema}} = e^{r/\tau}/Z(c) \le e^{(r_{\max} - b(c))/\tau}$ because $\tau \log Z(c) \ge b(c)$. Across intervals, let $\bar{\pi}_{\mathrm{ema}}^{(m)}$ denote the anchor of interval m, and suppose that each interval is solved exactly and the

anchor is replaced by the solution. Then $\bar{\pi}_{\mathrm{ema}}^{(m+1)} \propto \bar{\pi}_{\mathrm{ema}}^{(m)} e^{r/\tau}$ maximizes $\mathbb{E}_{\pi}[r] - \tau D_{\mathrm{KL}}(\pi \| \bar{\pi}_{\mathrm{ema}}^{(m)})$, i.e., it is a mirror-ascent step with step size $1/\tau$ on the expected reward, and M intervals give $\bar{\pi}_{\mathrm{ema}}^{(M)} \propto \bar{\pi}_{\mathrm{ema}}^{(0)} e^{Mr/\tau}$, the Gibbs policy of temperature τ/M relative to $\bar{\pi}_{\mathrm{ema}}^{(0)}$. The EMA update of Appendix C moves the anchor only 80% of the way in parameter space, so the practical recursion takes damped steps. These statements concern the exact-matching recursion; we make no claim of global convergence for the practical algorithm.

C Experimental Details

VIRM uses the ELBO-based ratio of Eq. (7) with $w(t) = 1$, $t \sim \mathcal{U}[0, 1]$, and one (t, ϵ) draw per replayed output. The anchor is updated at the end of each interval, i.e., every $U = 10$ training steps, as $\bar{\theta}_{\mathrm{ema}} \leftarrow 0.2\,\bar{\theta}_{\mathrm{ema}} + 0.8\,\theta$ (EMA decay rate 0.2). Table 2 lists the hyperparameters.

Table 2: VIRM hyperparameters.

Hyperparameter	SD3.5-Medium	FLUX.2-9B
Temperature τ	0.1	0.1
Group size G	8	8
Number of value heads H	4	4
Priority floor ε_p	10^{-4}	10^{-4}
Priority clip $p_{\max}$	1	1
EMA update frequency U (training steps)	10	10
EMA decay rate	0.2	0.2