Aligning Few-Step Generative Models by Amortizing Sample-based Variational Inference

Jaewoo Lee[*,1,2] **Hyeongyu Kang**[*,1] **Dohyun Kim**[1] **Kyuil Sim**[1]

Woocheol Shin[1] **Minsu Kim**[1,3] **Taeyoung Yun**[1] **Jeongjae Lee**[1]

Sanghyeok Choi[4] **Tabitha Edith Lee**[3,5] **Jong Chul Ye**[†,1] **Jinkyoo Park**[†,1,6]

[1]KAIST [2]MongooseAI [3]Mila – Quebec AI Institute
[4]University of Edinburgh [5]Université de Montréal [6]Omelet

{jaewoo, khg2000v}@kaist.ac.kr

Abstract

Aligning a few-step generative model is challenging, since existing alignment frameworks typically rely on restrictive assumptions: a tractable likelihood, a specific ODE/SDE solver, or a particular model family. We introduce **FAV** *(**F**ew-step Generative Models **A**lignment via Sample-based **V**ariational Inference)*, a general alignment framework that requires only sample access to the generator and the reference distribution. We cast alignment as sampling from a reward-tilted distribution anchored to a reference distribution. We leverage Stein Variational Gradient Descent as a sample-based variational inference scheme and amortize its particle updates into the generator parameters via fixed-point regression. We evaluate FAV on two domains: reinforcement learning and image generator alignment. On generative policy alignment for reinforcement learning, FAV outperforms prevailing policy extraction baselines across 56 offline and 30 offline-to-online RL tasks. For image generator alignment, FAV fine-tunes diverse few-step backbones, including GAN, drifting model, consistency models, and flow maps, scaling from ImageNet-256 to 1024^2 text-to-image synthesis. Code is available at `this link`.

1 Introduction

Recently emerging few-step generators [87, 26, 29, 113, 6, 17] are implicit models: they do not admit tractable likelihood evaluation [1], and their generation procedures are often tied to specialized samplers [87, 43, 17]. However, existing alignment methods for generative models rely on structural assumptions that few-step generators often do not satisfy: tractable likelihoods over denoising trajectories [4, 25, 97, 60, 49, 41, 99, 52, 115], rely on specific ODE/SDE solvers [21, 56, 105], or model-specific formulations [40, 19, 110, 104, 112].

To address this limitation, we propose ***F**ew-step Generative Models **A**lignment via Sample-based **V**ariational Inference* **(FAV)**. Our key design principle is to make the alignment procedure sample-based, thereby decoupling it from the generator family and sampling dynamics. FAV formulates alignment as sampling from the reward-tilted distribution:

$$q^*(x) \propto p_{\text{ref}}(x) \exp(r(x)),$$

[*]Equal contribution authors. [†]Corresponding authors.
Project page: `https://jaewoopudding.github.io/fav_project_page`

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

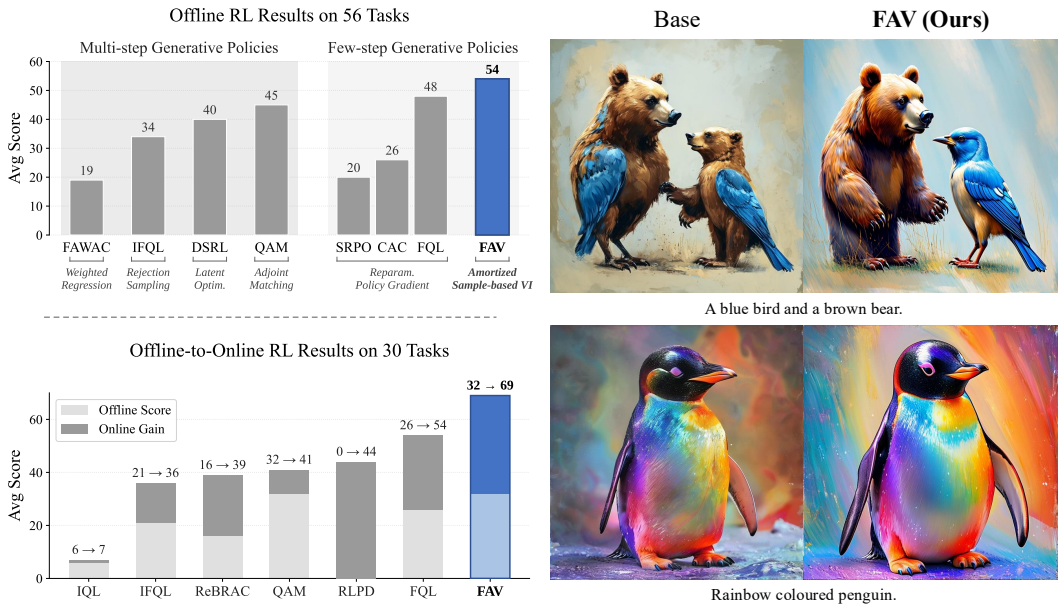

Figure 1: *Left*: Sampling from the Q-tilted distribution via FAV yields state-of-the-art performance on offline and offline-to-online RL tasks. *Right*: Sampling from a human-preference-tilted distribution through FAV improves image quality of a high-resolution text-to-image generator.

where p_{ref} denotes a reference distribution, such as a distribution induced by a pretrained generator or an empirical data distribution. FAV approximates q^* using Stein Variational Gradient Descent (SVGD) [59] and amortizes the resulting SVGD transport into the generator through fixed-point regression [100, 17]. Because SVGD requires the score of p_{ref}, which is intractable for empirical data and implicit generators, we estimate it nonparametrically with kernel density estimation (KDE) [74]. This construction requires only samples from the generative model and p_{ref}, therefore does not require assumptions such as tractable likelihoods, explicit sampling trajectories, or a particular model family. As a result, FAV can fine-tune diverse few-step generators, including VAEs [45], GANs [32], consistency models [88, 64, 63], and flow-map models [30, 6, 7]. FAV can also be used for pre-training by choosing the empirical data distribution as p_{ref}, yielding a sampler for the reward-tilted distribution without a pre-trained generator.

We evaluate FAV on reinforcement learning and image generator alignment tasks. In reinforcement learning, FAV achieves state-of-the-art performance on 56 offline RL tasks and 30 offline-to-online RL tasks from OGBench [71] and D4RL [27], outperforming prevailing policy extraction paradigms such as reparameterized policy gradient [72, 73]. In image generator alignment, we fine-tune few-step generators at two scales: at ImageNet-256 [16], we align StyleGAN-XL [82], drifting model [17], inductive moment matching [113], and iMeanFlow [30] to optimize an aesthetic reward [83]. We also generalize FAV to optimize a black-box reward by using zeroth-order gradient estimation. At the 1024^2 resolution text-to-image scale, we align SANA-Sprint [10] for a human preference score [101] and an NSFW safety detector [24]. Across both scales, FAV is agnostic to model architecture and sampler, and attains high reward while mitigating reward overoptimization.

2 Related works

Few-step generative models. GAN [32, 82] and VAE [45] are the earliest class of single-step deep generative models, but are notoriously unstable to train or tend to generate blurry samples. Following the advent of diffusion and flow models [88, 35, 55, 60], consistency models [87, 86, 31, 64] enable few-step generation by learning a consistency function that maps any point along a probability-flow ODE trajectory to its data point. Distillation-based approaches improve few-step diffusion generation by employing an adversarial objective [81] or distribution matching objective [108, 65]. Flow maps [43, 26, 29, 6, 7] instead learn the jump operator corresponding to ODE integration between two denoising steps, refining sample quality as the number of steps grows. More recently,

drifting model [17] learns one-step noise-to-data mappings by amortizing the transport induced by a kernel mean-shift drift field [11]. Despite this progress, pre-trained few-step generators can exhibit unintended behaviors, yet alignment methods tailored to few-step models remain underexplored.

Alignment of generative models. Generative model alignment can be pursued through two broad paradigms: maximizing the reward by seeking $\arg\max_{q_\theta} \mathbb{E}_{x\sim q_\theta}[r(x)]$, or sampling from the reward-tilted distribution $p_{\text{ref}}(x)\exp(r(x))$. For reward maximization, RL-based methods [4] and direct backpropagation approaches [14, 76] have been proposed. However, these methods often suffer from severe reward overoptimization, leading to degraded sample quality and reduced diversity. Sampling from a reward-tilted distribution mitigates overoptimization by anchoring the optimization to the p_{ref}. To sample from the reward-tilted distribution, prior work has explored KL-regularized RL defined on diffusion denoising Markov decision process (MDP) [25, 94, 93, 53, 56, 41, 49], sampling-based methods [85, 44, 111], guidance methods [12, 62, 37, 36], and stochastic optimal control [21, 50]. Despite the substantial success of reward-tilted sampling, extending this success to few-step generative models remains underexplored.

3 Preliminaries

3.1 Alignment as sampling from a reward-tilted distribution

Given a reference distribution $p_{\text{ref}}(x)$ and an objective function $r(x)$ (e.g., a reward model or Q-function), we cast the alignment problem as sampling from the unnormalized density:

$$q^*(x) \propto p_{\text{ref}}(x)\exp\left(\beta \cdot r(x)\right), \tag{1}$$

where $\beta > 0$ is a temperature parameter governing the strength of the alignment [44]. This formulation naturally emerges from Variational Inference (VI), which is equivalent to the KL-regularized reinforcement learning objective [39, 102, 46, 25]. The optimal variational distribution q^* is the minimizer of the following KL objective:

$$q^*(x) = \arg\min_q D_{\text{KL}}\left(q(x) \,\middle\|\, \frac{1}{Z}p_{\text{ref}}(x)\exp\left(\beta \cdot r(x)\right)\right), \tag{2}$$

where the Z is a normalizing constant. Sampling from $q^*(x)$ can yield high-reward samples while preserving the property of the reference distribution [93, 95]. This perspective has led to two primary application domains: conservative policy extraction in reinforcement learning (RL) and reward alignment of image generators. In offline RL, approximating $q^*(x)$ via weighted behavior cloning [40, 19, 110], GFlowNet [97], guidance [62], and Adjoint Matching [50] have shown strong performance on challenging reinforcement learning tasks. In image generation, reward-weighted flow matching [104, 112], entropy-regularized RL [99, 56, 94, 41], stochastic optimal control [21], and probabilistic inference [44, 49] have been successfully applied to align image generators.

3.2 Stein Variational Gradient Descent

Standard VI objectives typically require density evaluation of $q(x)$ [78, 5], which is intractable when $q(x)$ is induced by an implicit generator. SVGD [59, 57] bypasses this requirement by representing $q(x)$ with particles and iteratively transporting them toward the target distribution $p(x)$:

$$x_i^{\ell+1} \leftarrow x_i^\ell + \epsilon\phi^*_{q_\ell,p}(x_i), \quad \forall i = 1,\cdots,n, \qquad \phi^*_{q_\ell,p} = \arg\max_{\phi\in\mathcal{F}}\left\{-\frac{d}{d\epsilon}\text{KL}(q_\ell \,\|\, p)\Big|_{\epsilon=0}\right\}. \tag{3}$$

Here, q_ℓ denotes the empirical distribution of the particles at the ℓ-th iteration, ϵ denotes the step size, and $\phi^*_{q_\ell,p}(x)$ represents the optimal Stein velocity field that maximizes the decreasing rate of the KL divergence between q_ℓ and the target p. When function space $\mathcal{F}$ is restricted to be the unit ball of a reproducing kernel Hilbert space (RKHS) associated with a positive definite kernel $k(x,x')$, the gradient of the KL divergence in Equation (3) leads to the closed-form solution [59]:

$$\phi^*_{q_\ell,p}(\cdot) \propto \mathbb{E}_{x\sim q_\ell}[\mathcal{A}_p k(x,\cdot)] = \mathbb{E}_{x\sim q_\ell}[k(x,\cdot)\,\nabla_x \log p(x) + \nabla_x k(x,\cdot)], \tag{4}$$

where $\mathcal{A}_p$ is a linear operator called Stein operator. Therefore, SVGD iteratively pushes particles using the optimal gradient direction ϕ^*, and this eventually converges to the target p with sufficiently small ϵ, under which $\phi^*_{q_\infty,p} \equiv 0$ (i.e. $q_\infty = p$ if and only if $\phi^*_{q_\infty,p} \equiv 0$ when $k(x,x')$ is strictly positive definite in an appropriate sense [58, 69, 13] such as the RBF kernel).

4 Challenges in aligning few-step generative models

Aligning few-step generative models has a key desideratum: improving sample quality while preserving fast generation. Under this advantage, few-step generative policy alignment can yield expressive policies with single-step inference, avoiding backpropagation through time and the credit-assignment challenges of multi-step generative policies. In image generator alignment, it enables high-reward generation without sacrificing the efficiency of few-step generators. However, existing alignment methods do not naturally extend to few-step generative models. To clarify this limitation, we categorize few-step generators into two classes: **(i) one-step noise-to-data mappings** and **(ii) flow maps**, and examine how the existing methods break down in each case.

One-step noise-to-data mappings. VAE [45], GAN [32], and Drifting Models [17] map latent noise directly to data in a single feed-forward pass. They do not expose the denoising trajectory required by prior alignment methods: RL-based approaches assume a sequential denoising process [25, 4, 56] or a flow matching backbone [104, 112], whereas SOC-based methods assume explicit ODE/SDE dynamics [21, 61]. Moreover, directly measuring the KL divergence for regularization may require additional effort because it requires explicit likelihood evaluation.

Flow maps. Recent emergent flow map generators [26, 113, 6, 7, 29], including consistency models [87, 43, 63], learn to jump between two noise levels. Flow map inference is performed using learned jump operators between two timesteps rather than numerically integrating the underlying ODE or SDE. Specifically, multistep consistency sampling, adopted by consistency models [87, 64, 86, 63, 31] and distilled diffusion models [81, 108, 107], generates samples through a discrete iteration of consistency-map evaluations and perturbation with i.i.d. sampled Gaussian noise. Therefore, applying alignment methods built upon SDE/ODE solvers is not straightforward for flow map models.

5 Aligning few-step generative models via sample-based VI

In this section, we introduce FAV, a sample-based alignment framework for few-step generative models. Our key design principle is to build a sample-based method, making the alignment process agnostic to the generator family and sampling procedure. A sample-based approach is well-suited to few-step generators, whose low per-sample cost makes sample-based alignment efficient. Our method consists of three steps: (i) approximating the reward-tiled distribution via sample-based SVGD, (ii) estimating the intractable reference score through KDE, and (iii) amortizing the resulting Stein transport into the generator to preserve fast generation.

5.1 Stein velocity field towards tilted distribution

Let $f_\theta : \mathcal{Z} \to \mathcal{X}$ be a neural network parameterized mapping function that transforms a latent variable z drawn from a prior $\mathcal{Z}$ into the data sample x of $\mathcal{X}$, and q_θ be the empirical distribution of $f_\theta(\epsilon)$ with $\epsilon \sim p_{\text{noise}}$. SVGD offers a sample-based method for approximating the reward-tilted distribution $q^*(x)$ by only requiring samples from the implicit generator $f_\theta(\epsilon)$. By substituting the target distribution with the reward-tilted distribution, we derive the optimal Stein velocity field $\phi^*_{q_\theta, q^*}$ that drives samples from the current model q_θ toward q^*:

$$\phi^*_{q_\theta,q^*}(x) = \mathbb{E}_{x'\sim q_\theta}\Big[\underbrace{k(x',x)\nabla_{x'}\log p_{\text{ref}}(x')}_{\text{Prior Alignment}} + \underbrace{\beta\cdot k(x',x)\nabla_{x'}r(x')}_{\text{Reward Guidance}} + \underbrace{\nabla_{x'}k(x',x)}_{\text{Diversity Enforcement}}\Big]. \quad (5)$$

Note that the reward gradient can be computed directly from a differentiable reward function, or estimated using zeroth-order methods for black-box rewards. See Appendix C for details about zeroth-order gradient estimation.

5.2 KDE for intractable score estimation

Since $\nabla \log p_{\text{ref}}$ is typically intractable, we estimate it nonparametrically via kernel density estimation (KDE) [54, 114, 88]. Instantiating both the KDE and SVGD kernel to be Gaussian RBFs k_σ with

Algorithm 1. FAV Loss.

```
# p_ref  : dataset / pre-trained model
# r      : reward function

eps      = randn([N_gen, C])
x_ref    = sample_from(p_ref, N_ref)
x        = generator(eps)

phi      = get_stein_transport(x, x_ref, r)
x_target = stopgrad(x + phi)

loss     = mse_loss(x, x_target)
```

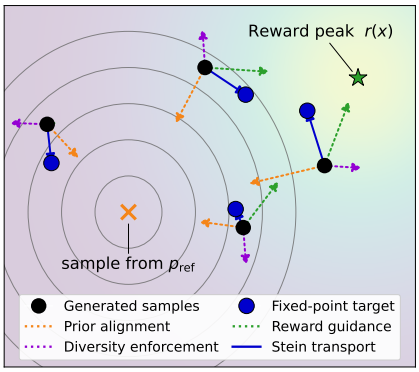

Figure 2: Illustration of SVGD transport.

shared bandwidth σ yields the following approximated transport field:

$$\hat{\phi}^*_{q_\theta, q^*_\sigma}(x) = \mathbb{E}_{\substack{x' \sim q_\theta \\ x^{\text{ref}} \sim p_{\text{ref}}}} \left[k_\sigma(x', x) \left(\frac{\tilde{k}_\sigma(x', x^{\text{ref}})(x^{\text{ref}} - x')}{\sigma^2} + \beta \cdot \nabla_{x'} r(x') + \frac{(x - x')}{\sigma^2} \right) \right], \quad (6)$$

where $\tilde{k}_\sigma(x', x^{\text{ref}}) := \frac{k_\sigma(x', x^{\text{ref}})}{\mathbb{E}_{\bar{x}^{\text{ref}}}[k_\sigma(x', \bar{x}^{\text{ref}})]}$. The resulting velocity field $\hat{\phi}^*_{q_\theta, q^*_\sigma}$ pushes samples toward a surrogate posterior $q^*_\sigma(x) \propto p_\sigma(x) \exp(\beta \cdot r(x))$, where $p_\sigma = p_{\text{ref}} * k_\sigma$. We establish the consistency conditions under which $q^*_\sigma \to q^*$ and a detailed derivation of Equation (6) in Appendix B. The choice of p_{ref} determines the training regime. When p_{ref} is the empirical data distribution, FAV directly learns an aligned generator from data without a separate reference generator. When p_{ref} is the distribution of a pre-trained model, FAV fine-tunes that generator.

5.3 Amortization via fixed-point regression

While iterative SVGD updates of the form $x' \leftarrow x + \hat{\phi}^*_{q_\theta, q^*_\sigma}(x)$ in Equation (6) can approximate the reward-tilted distribution, repeated transport incurs inference-time overhead that erodes the core advantage of few-step generative models: fast sampling. Therefore, we amortize the transport via the Stein velocity field into the network parameters θ through a fixed-point regression [100, 17, 48]:

$$\mathcal{L}(\theta) = \mathbb{E}_{\epsilon \sim \mathcal{N}}[||\text{stopgrad}(f_\theta(\epsilon) + \hat{\phi}^*_{q_\theta, q^*_\sigma}(f_\theta(\epsilon))) - f_\theta(\epsilon)||_2^2]. \quad (7)$$

The stop-gradient operator casts the transported sample as a stable regression target. Minimizing this objective distills the SVGD update into the generator, moving the model distribution q_θ toward the smoothed reward-tilted target q^*_σ while preserving the advantage of few-step sampling.

5.4 FAV in pre-trained representation space

For high-dimensional data such as images, kernels on raw spatial distances often fail to capture perceptually meaningful similarity between data points [33, 109]. Motivated by kernel-based distribution matching in learned representation spaces [3, 17], we perform FAV training in the representation space of the pre-trained encoder ψ rather than in the raw space. Our amortization objective on the representation space is given by:

$$\mathcal{L}(\theta) = \mathbb{E}_{\epsilon \sim \mathcal{N}}[||\text{stopgrad}(\psi(f_\theta(\epsilon)) + \hat{\phi}^*_{q_\theta, q^*_\sigma}(\psi(f_\theta(\epsilon)))) - \psi(f_\theta(\epsilon))||_2^2]. \quad (8)$$

6 Experiments

In this section, we evaluate FAV on three domains: 2D toy setup, policy alignment for reinforcement learning, and image generator alignment. Through these experiments, we demonstrate that FAV is a domain-agnostic and powerful method for sampling from reward-tilted distributions, applicable from low-dimensional data, proprioceptive data, to high-resolution 1024^2 images. Moreover, we demonstrate that FAV generalizes across diverse classes of few-step generators, including single-step mappings such as VAE and GAN, as well as consistency models and flow maps. We provide the ablation and sensitivity analysis in Appendix F and Appendix G, respectively.

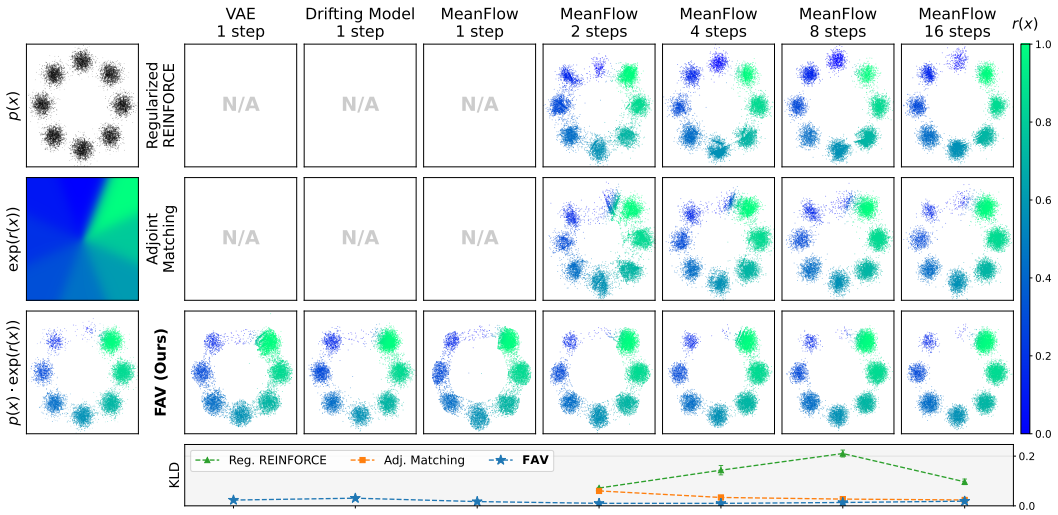

Figure 3: **Sampling from a reward-tilted distribution.** On the 8 Gaussian $p(x)$ with the reward function shown on the left, we target to sample from $p(x)\exp(r(x))$. Regularized REINFORCE and Adjoint Matching are not applicable to 1-step generators, whereas FAV applies uniformly across all architectures. For MeanFlow from 2 to 16 sampling steps, FAV consistently yields samples that better match the reward-tilted target and achieves lower KL divergence than baselines.

6.1 Toy setting

To build intuition for the ability of FAV to sample from reward-tilted distributions, we conduct a toy study on an 8-mode 2D Gaussian mixture. First, we pre-train three representative few-step generators: VAE [45], drifting model [17], and MeanFlow [29]. Then, we fine-tune each model with FAV and two prevailing alignment baselines: policy gradient [91] on a denoising MDP [4] and Adjoint Matching [21]. Following [97, 15], we regularize the policy gradient with an approximate KL penalty to sample from the tilted distribution. Because baselines require SDE integration for sampling, we treat the average velocity of MeanFlow as an instantaneous velocity and inject noise at each denoising step. See Appendix A.1 for details.

As shown in Figure 3, FAV consistently attains the lowest KL divergence to the target distribution across all settings. Policy gradient and Adjoint Matching are either inapplicable or non-trivial to adapt to the VAE, Drifting Model, and 1-step MeanFlow. Notably, both baselines degrade significantly in the few-step regime since they rely on alignment signals defined over multi-step denoising trajectories, which become ill-suited when the sampling process is collapsed into only a few steps. Overall, these results demonstrate that FAV is a strong, model-agnostic choice for aligning few-step generators.

6.2 Aligning generative policy with FAV

Sampling actions from the Q-tilted distribution: $p_{\text{data}}(a \mid s)\exp(Q(s,a))$ is a promising approach for offline RL and offline-to-online RL [75, 102, 66, 97, 110]. Our RL experiments assess two aspects of FAV: **(1) policy extraction capability** relative to prior policy extraction methods, and **(2) transition ability from offline to online learning**.

6.2.1 Experimental setup

Benchmarks. We evaluate FAV on 56 offline RL tasks: 50 reward-based variants (`-singletask`) from OGBench [71], spanning 10 environments with diverse embodiments and control challenges, and 6 challenging antmaze tasks from D4RL [27]. For offline-to-online RL, we select the 30 OGBench tasks across six environments on which FAV's offline performance is weakest. We run 8 seeds and report the mean and the 95% confidence interval.

FAV configuration. We instantiate FAV as a single-step noise-to-action generator $\pi_\theta(a \mid s, z) : \mathbb{R}^s \times \mathbb{R}^a \to \mathbb{R}^a$, avoiding backpropagation through time and simplifying credit assignment relative to multi-step generative policies. We set an offline dataset as the reference distribution p_{data} to

Table 1: **Offline RL performance.** FAV achieves the best overall average performance among all compared methods across 56 tasks from OGBench [71] and D4RL [27]. FAV-Adaptive, with pre-calculated bandwidth through data, also outperforms all baselines in average performance.

	Gaussian Policies		Latent Opt.	Flow-based Policies			Distillation Policies			**Ours**	
Task Categories (offline 1M)	IQL	ReBRAC	DSRL	FAWAC	IFQL	QAM	SRPO	CAC	FQL	**FAV-Adaptive**	**FAV**
OGBench antmaze-large-navigate-singletask (5 tasks)	53 ± 3	81 ± 5	40 ± 29	6 ± 1	28 ± 5	77 ± 5	11 ± 4	33 ± 4	79 ± 3	80 ± 2	**87** ± 2
OGBench antmaze-giant-navigate-singletask (5 tasks)	4 ± 1	**26** ± 8	0 ± 0	0 ± 0	3 ± 2	3 ± 1	0 ± 0	0 ± 0	9 ± 6	**26** ± 3	**26** ± 5
OGBench humanoidmaze-medium-navigate-singletask (5 tasks)	33 ± 2	22 ± 8	34 ± 20	19 ± 1	60 ± 14	63 ± 2	1 ± 1	53 ± 8	58 ± 5	44 ± 8	**64** ± 10
OGBench humanoidmaze-large-navigate-singletask (5 tasks)	2 ± 1	2 ± 1	**10** ± 12	0 ± 0	**11** ± 2	4 ± 3	0 ± 0	0 ± 0	4 ± 2	4 ± 1	5 ± 3
OGBench antsoccer-arena-navigate-singletask (5 tasks)	8 ± 2	0 ± 0	28 ± 9	12 ± 0	33 ± 6	**61** ± 1	1 ± 0	2 ± 4	60 ± 2	60 ± 2	59 ± 2
OGBench cube-single-play-singletask (5 tasks)	83 ± 3	91 ± 2	93 ± 14	81 ± 4	79 ± 2	57 ± 12	80 ± 5	85 ± 9	**96** ± 1	92 ± 1	93 ± 1
OGBench cube-double-play-singletask (5 tasks)	7 ± 1	12 ± 1	**53** ± 14	5 ± 2	14 ± 3	30 ± 7	2 ± 1	6 ± 2	29 ± 2	24 ± 3	26 ± 2
OGBench scene-play-singletask (5 tasks)	28 ± 1	41 ± 3	**88** ± 9	30 ± 3	30 ± 3	59 ± 0	20 ± 1	40 ± 7	56 ± 2	55 ± 1	55 ± 1
OGBench puzzle-3x3-play-singletask (5 tasks)	9 ± 1	21 ± 1	0 ± 0	6 ± 2	19 ± 1	19 ± 7	18 ± 1	19 ± 0	30 ± 1	63 ± 8	**73** ± 8
OGBench puzzle-4x4-play-singletask (5 tasks)	7 ± 1	14 ± 1	37 ± 13	1 ± 0	25 ± 5	**38** ± 3	10 ± 3	15 ± 3	17 ± 2	12 ± 2	16 ± 5
D4RL antmaze (6 tasks)	57	78	56 ± 2	44 ± 3	65 ± 7	80 ± 10	74	30 ± 3	**84** ± 3	79 ± 1	80 ± 6
Total Average (56 tasks)	27	36	40	19	34	45	20	26	48	50	**54**

Table 2: **Offline-to-online RL performance.** We consider the 30 OGBench tasks from the six environments where FAV attains the lowest average performance. After an additional 1M online steps, FAV achieves the best post-training performance.

Task Categories (offline 1M → online 1M)	IQL	ReBRAC	RLPD	IFQL	QAM	FQL	FAV
antmaze-giant-navigate-singletask (5 tasks)	4 ± 2 → 3 ± 2	32 ± 7 → **99** ± 1	0 ± 0 → 64 ± 15	2 ± 2 → 0 ± 0	4 ± 1 → 11 ± 6	0 ± 0 → 44 ± 10	28 ± 5 → 74 ± 3
humanoidmaze-large-navigate-singletask (5 tasks)	2 ± 1 → 2 ± 1	1 ± 2 → 0 ± 0	0 ± 0 → 0 ± 0	12 ± 2 → 12 ± 4	4 ± 3 → **21** ± 7	5 ± 2 → 4 ± 6	6 ± 3 → 8 ± 6
antsoccer-arena-navigate-singletask (5 tasks)	8 ± 3 → 4 ± 1	0 ± 0 → 0 ± 0	0 ± 0 → 37 ± 4	27 ± 11 → 49 ± 10	61 ± 4 → 93 ± 5	57 ± 5 → 89 ± 5	62 ± 4 → **92** ± 1
cube-double-play-singletask (5 tasks)	2 ± 1 → 0 ± 0	6 ± 2 → 35 ± 9	0 ± 0 → 2 ± 2	9 ± 1 → 54 ± 6	26 ± 11 → 24 ± 21	27 ± 2 → 75 ± 5	25 ± 5 → **91** ± 2
scene-play-singletask (5 tasks)	21 ± 3 → 33 ± 6	41 ± 4 → 60 ± 0	0 ± 0 → 59 ± 0	47 ± 5 → 57 ± 2	59 ± 1 → 60 ± 0	55 ± 2 → 60 ± 0	54 ± 2 → **62** ± 5
puzzle-4x4-play-singletask (5 tasks)	2 ± 1 → 0 ± 0	13 ± 2 → 38 ± 7	0 ± 0 → **100** ± 0	29 ± 4 → 45 ± 9	38 ± 4 → 40 ± 17	12 ± 3 → 50 ± 10	18 ± 5 → 85 ± 9
Total Average (30 tasks)	6 ± 1 → 7 ± 1	16 ± 1 → 39 ± 2	0 ± 0 → 44 ± 3	21 ± 2 → 36 ± 3	32 ± 2 → 41 ± 6	26 ± 1 → 54 ± 3	32 ± 2 → **69** ± 3

pre-train the agent directly on offline data. Additionally, we propose **FAV-Adaptive**, which sets σ automatically via Scott's rule [84]. We adopt the same actor and critic architectures as in FQL [73] and use the same optimization hyperparameters across all baselines to ensure a fair comparison.

Baselines. We compare FAV against 10 representative policy extraction methods in both offline RL and offline-to-online RL. For offline RL, we consider four categories of baselines: **(1) Gaussian policies.** IQL [47] and ReBRAC [92]. In particular, ReBRAC targets the Q-tilted distribution and produces actions in a single forward pass, analogous to FAV. **(2) Latent-space policies.** DSRL [98], which acts in the latent space of a pre-trained flow policy. While it is agnostic to policy type, it introduces additional inference-time overhead. **(3) Multi-step flow policies.** We compare FAV with FAWAC, IFQL [34], and QAM [50], all of which require iterative denoising. Each of them extracts policy from value function via weighted behavioral cloning [66], rejection sampling, and Adjoint Matching [21], respectively. **(4) Few-step distilled policies.** SRPO [9], CAC [19], and FQL [73] distill a generative model into a one- or few-step model: SRPO [9] distills a diffusion score, CAC applies consistency training [87], and FQL distills from flow-matching behavioral policy. For offline-to-online RL, we additionally include RLPD [2], a dedicated offline-to-online method.

6.2.2 Results

FAV outperforms baselines on both offline and offline-to-online RL. Table 1 reports results on 56 offline RL tasks across 50 OGBench tasks and 6 D4RL antmaze tasks. FAV achieves the best overall average performance, outperforming 9 competitive baselines across diverse reinforcement learning tasks. Despite being a single-step generator without an auxiliary generative model, FAV outperforms two baseline categories: multi-step generative policies (FAWAC, IFQL, QAM), which require iterative denoising, and distilled generative policies (SRPO, FQL), which are built on auxiliary generative models. Additionally, FAV surpasses DSRL, which additionally incurs inference-time overhead. The results show that FAV-Adaptive also surpasses all prior baselines on average. This makes FAV-Adaptive a strong default and a reliable starting point for hyperparameter search. The details of adaptive bandwidth configuration are elaborated in Appendix A.2. For offline-to-online RL, Table 2 demonstrates that FAV achieves the best post-adaptation performance, showing that the objective remains effective in online adaptation.

We attribute the strong performance of FAV to three factors: FAV can utilize a powerful gradient signal of the Q-function as in reparameterized policy gradients [72]; FAV framework enables a single-step policy, thereby avoiding the credit-assignment and backpropagation-through-time challenges of multi-step generative policies; and FAV does not rely on a complex auxiliary generative model or its distillation process that may harm the training stability and efficiency [20, 9, 73].

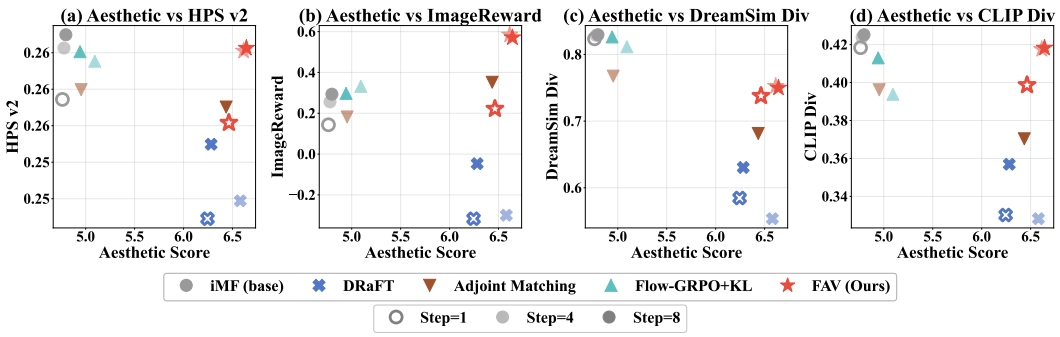

Figure 4: **Target reward vs. evaluation metrics.** Aesthetic Score is the target reward; (a) HPSv2, (b) ImageReward, (c) DreamSim diversity and (d) CLIP diversity evaluate quality and diversity to indicate reward overoptimization. FAV achieves the best Pareto frontier.

Table 3: **Comparison of inference-time alignment methods with FAV.** FAV can generate aligned samples with substantially lower inference cost than inference-time baselines.

Method	Time (s) ↓	Aesthetic Score ↑	HPSv2 ↑	ImageReward ↑	DreamSim Div ↑	CLIP Div ↑
BoN-256	28.6	5.68 (0.00)	**26.23 (0.00)**	0.44 (0.00)	**0.82 (0.00)**	0.41 (0.00)
ReNO-50	17.9	6.52 (0.01)	26.01 (0.00)	0.31 (0.03)	0.80 (0.00)	**0.45 (0.00)**
FAV (ours)	**0.1**	**6.61 (0.09)**	26.01 (0.00)	**0.58 (0.07)**	0.75 (0.01)	0.42 (0.01)

FAV as an attractive option for policy extraction. The results suggest that FAV is an attractive option for policy extraction, competitive with strong representative approaches such as reparameterized policy gradients [72, 73] and Adjoint Matching [50]. The comparison with FQL is particularly clean: FAV and FQL share the same codebase, model architectures, and single-step inference, differing only in the extraction objective. FQL relies on reparameterized policy gradients and an auxiliary flow-matching model for distillation; FAV directly amortizes sample-based variational inference. Despite this simpler mechanism, FAV outperforms FQL in both offline and offline-to-online RL, highlighting FAV as an effective class of policy extraction methods for RL.

6.3 Conditional image generation

Sampling from the reward-tilted distribution $p_{\text{ref}}(x)\exp(r(x))$ is a principled approach to image alignment, yielding high-reward samples while mitigating reward overoptimization [93]. In this section, we demonstrate three key advantages of FAV: (1) superior alignment performance over existing baselines, (2) the preservation of fast generation speeds, and (3) generalizability across diverse model classes and reward settings. We provide implementation details in Appendix A.3.

6.3.1 Experimental setup

ImageNet-256 We fine-tune StyleGAN-XL [82], iMM [113], iMeanFlow [30], and drifting model [17] to optimize the LAION aesthetic score [83]. To detect reward over-optimization, we measure the image quality using Human Preference Score (HPSv2) [101] and ImageReward [103], and diversity using mean pairwise cosine distance of DreamSim features (DreamSim Diversity) [28], and CLIP features (CLIP Diversity) [77]. We compare FAV against baseline alignment methods on iMeanFlow with step sizes of $\{1, 4, 8\}$.

High-resolution text-to-image generation We fine-tune Sana-sprint 1.6B [10] and evaluate on two distinct alignment tasks: (1) safety alignment using Falconsai NSFW image classifier [24] as the negative reward on SneakyPrompt [106] and (2) human preference alignment, using HPSv2 as the target reward on DrawBench [80]. To indicate overoptimization, we employ ImageReward, Dreamsim Diversity, and CLIP Diversity. We conduct experiments with a step size of 4.

Baselines We compare FAV against 4 categories of generative model alignment methods. **(1) Direct backpropagation**: DRaFT [14] leverages reward gradients directly, which is efficient for few-step implicit models due to short gradient chains. **(2) RL-based methods**: Flow-GRPO [56] converts the

deterministic Flow-ODE into a marginal-preserving SDE to measure transition likelihoods, enabling the use of the policy gradient method. RLCM [70] applies policy gradient over the multistep sampling trajectory of consistency models [87, 63]. **(3) SoC-based methods**: Adjoint Matching [21] converts the ODE into an SDE with memoryless noise schedules and uses the lean adjoint regression loss. **(4) Inference-time alignment methods**: Best-of-N [42] selects the highest-reward sample from N candidates, and ReNO [23] optimizes the noise space using the reward gradient.

Evaluation In the ImageNet-256 experiments, we randomly select 32 class labels from the full set of ImageNet classes and use them for both training and evaluation. In the text-to-image experiments, we randomly sample 32 prompts from SneakyPrompt and DrawBench to form the evaluation set, while training is conducted exclusively on the remaining prompts. For every metric, we generate 32 images for each of the 32 evaluation labels/prompts, and compute the metrics over the resulting 1,024 images. All experiments are conducted across 4 random seeds.

6.3.2 Results

FAV outperforms baselines in few-step regimes. Figure 4 shows that FAV achieves the best Pareto frontier across {1,4,8} generation steps. DRaFT improves the target reward at all step sizes, but substantially degrades quality and diversity metrics. Adjoint Matching is competitive at 8-steps, but becomes less effective with fewer steps and is not applicable to the 1-step setting, consistent with Section 6.1. Notably, 1-step FAV already outperforms or matches 8-step baselines, demonstrating its effectiveness in the few-step regime. Qualitative comparisons are provided in Figure 12.

FAV preserves fast generation. As shown in Table 3, FAV generates samples approximately $180\times$ to $280\times$ faster than inference-time alignment methods with competitive image quality and diversity. By amortizing SVGD transport, FAV fully preserves the inference efficiency of few-step generative models.

Table 4: **FAV across diverse generator classes.** Evaluation metrics are reported at a comparable aesthetic score for a fair comparison.

Model	Aesthetic ↑	HPSv2 ↑	Dreamsim ↑
StyleGAN-XL(DRaFT) [82]	4.61 → 5.68	25.28 → 24.74	0.81 → 0.67
StyleGAN-XL(FAV) [82]	4.61 → **5.69**	25.28 → **25.14**	0.81 → **0.70**
Drifting(DRaFT) [17]	4.71 → 6.20	25.36 → 24.37	0.82 → 0.56
Drifting(FAV) [17]	4.71 → **6.27**	25.36 → **24.89**	0.82 → **0.68**
iMM(DRaFT) [113]	4.73 → 6.17	25.67 → 24.98	0.83 → 0.67
iMM(FAV) [113]	4.73 → **6.26**	25.67 → **25.55**	0.83 → **0.75**
iMeanFlow(DRaFT) [30]	4.80 → 6.58	26.14 → 25.00	0.83 → 0.55
iMeanFlow(FAV) [30]	4.80 → **6.61**	26.14 → **26.01**	0.83 → **0.75**

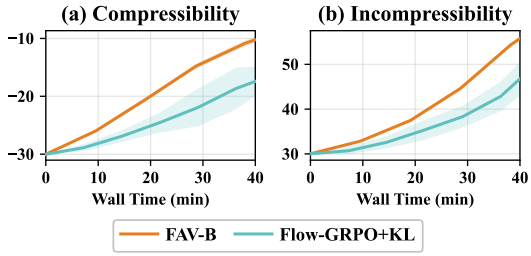

Figure 5: **FAV-B on black-box reward functions.**

FAV generalizes to diverse generators and reward functions. We evaluate FAV on one-step noise-to-data mapping models (StyleGAN-XL, Drifting model), and 4-step flow map models (iMM, iMeanFlow). Among the fine-tuning baselines, DRaFT is the only method that can handle all model types, yet it cannot mitigate reward overoptimization. Table 4 shows that FAV consistently outperforms DRaFT in both target reward and evaluation metrics across all settings. Furthermore, as demonstrated in Figure 11, FAV outperforms the baselines on SANA-Sprint 1.6B alignment at 1024^2 resolution, showing its scalability. We provide qualitative results in Figure 13.

FAV-B for black-box reward functions. We propose **FAV-B** to estimate the reward gradient with zeroth-order methods (See details in Appendix C). As shown in Figure 5, FAV-B effectively optimizes the compressibility and incompressibility rewards [4] and outperforms Flow-GRPO+KL in aligning the 4-step iMeanFlow model, while achieving approximately $1.5\times$ faster training.

7 Discussion

Limitations. FAV employs KDE-based nonparametric score estimation, whose consistency is guaranteed in the asymptotic regime. Hence, finite reference samples may introduce approximation error, particularly in high-dimensional domains. We mitigate this in image experiments by performing transport in a pretrained representation space, but this does not fully resolve the approximation error in high-dimensional data. Still, FAV consistently improves alignment across toy, generative policy, and image-generation settings, showing that the sample-based approximation is effective in practice.

Conclusion. We introduce FAV, a novel sample-based alignment framework for few-step generative models. We approximate a reward-tilted distribution using SVGD with KDE, and amortize the resulting Stein transport into the generator to preserve fast generation. We evaluated FAV on two domains: generative policy alignment and image generator alignment. In reinforcement learning, FAV outperformed multi-step flow policies and distilled policies on 56 offline RL tasks and 30 offline-to-online RL tasks, highlighting its effectiveness for policy extraction. In image alignment, FAV consistently achieved strong alignment performance while preserving sample quality and diversity across resolutions from 256^2 to 1024^2.

Acknowledgement

We thank the anonymous reviewers for their insightful comments and suggestions, which significantly improved our manuscript. This work is supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2022-NR068758, No. RS-2024-00410082, No. RS-2024-00336454, No. RS-2025-00563763). This work was also supported by the IVADO Postdoctoral Research Funding Program.

References

[1] Xinyue Ai, Yutong He, Albert Gu, Ruslan Salakhutdinov, J Zico Kolter, Nicholas Matthew Boffi, and Max Simchowitz. Joint distillation for fast likelihood evaluation and sampling in flow-based models. *arXiv preprint arXiv:2512.02636*, 2025.

[2] Philip J Ball, Laura Smith, Ilya Kostrikov, and Sergey Levine. Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning*, pages 1577–1594. PMLR, 2023.

[3] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018.

[4] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *The Twelfth International Conference on Learning Representations*.

[5] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[6] Nicholas Matthew Boffi, Michael Samuel Albergo, and Eric Vanden-Eijnden. Flow map matching with stochastic interpolants: A mathematical framework for consistency models. *Transactions on Machine Learning Research*.

[7] Nicholas Matthew Boffi, Michael Samuel Albergo, and Eric Vanden-Eijnden. How to build a consistency model: Learning flow maps via self-distillation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[8] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Yash Katariya, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

[9] Huayu Chen, Cheng Lu, Zhengyi Wang, Hang Su, and Jun Zhu. Score regularized policy optimization through diffusion behavior. In *The Twelfth International Conference on Learning Representations*.

[10] Junsong Chen, Shuchen Xue, Yuyang Zhao, Jincheng Yu, Sayak Paul, Junyu Chen, Han Cai, Song Han, and Enze Xie. Sana-sprint: One-step diffusion with continuous-time consistency distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16185–16195, 2025.

[11] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE transactions on pattern analysis and machine intelligence*, 17(8):790–799, 1995.

[12] Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*.

[13] Kacper Chwialkowski, Heiko Strathmann, and Arthur Gretton. A kernel test of goodness of fit. In *International conference on machine learning*, pages 2606–2615. PMLR, 2016.

[14] Kevin Clark, Paul Vicol, Kevin Swersky, and David J Fleet. Directly fine-tuning diffusion models on differentiable rewards. In *The Twelfth International Conference on Learning Representations*.

[15] Tristan Deleu, Padideh Nouri, Yoshua Bengio, and Doina Precup. Relative trajectory balance is equivalent to trust-pcl. *arXiv preprint arXiv:2509.01632*, 2025.

[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[17] Mingyang Deng, He Li, Tianhong Li, Yilun Du, and Kaiming He. Generative modeling via drifting. *arXiv preprint arXiv:2602.04770*, 2026.

[18] Luc Devroye and László Györfi. *Nonparametric Density Estimation: The L1 View*. John Wiley & Sons, 1985.

[19] Shutong Ding, Ke Hu, Zhenhao Zhang, Kan Ren, Weinan Zhang, Jingyi Yu, Jingya Wang, and Ye Shi. Diffusion-based reinforcement learning via q-weighted variational policy optimization. *Advances in Neural Information Processing Systems*, 37:53945–53968, 2024.

[20] Zihan Ding and Chi Jin. Consistency models as a rich and efficient policy class for reinforcement learning. In *The Twelfth International Conference on Learning Representations*.

[21] Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky TQ Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. In *The Thirteenth International Conference on Learning Representations*.

[22] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural networks*, 107:3–11, 2018.

[23] Luca Eyring, Shyamgopal Karthik, Karsten Roth, Alexey Dosovitskiy, and Zeynep Akata. Reno: Enhancing one-step text-to-image models through reward-based noise optimization. *Advances in Neural Information Processing Systems*, 37:125487–125519, 2024.

[24] Falcons.ai. Nsfw image detection model. `https://huggingface.co/Falconsai/nsfw_image_detection`, 2024. Accessed: 2025-10-09.

[25] 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.

[26] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *The Thirteenth International Conference on Learning Representations*.

[27] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[28] Stephanie Fu, Netanel Y Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. Dreamsim: learning new dimensions of human visual similarity using synthetic data. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pages 50742–50768, 2023.

[29] Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[30] Zhengyang Geng, Yiyang Lu, Zongze Wu, Eli Shechtman, J Zico Kolter, and Kaiming He. Improved mean flows: On the challenges of fastforward generative models. *arXiv preprint arXiv:2512.02012*, 2025.

[31] Zhengyang Geng, Ashwini Pokle, Weijian Luo, Justin Lin, and J Zico Kolter. Consistency models made easy. In *The Thirteenth International Conference on Learning Representations*.

[32] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[33] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[34] Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

[35] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[36] Peter Holderrieth, Douglas Chen, Luca Eyring, Ishin Shah, Giri Anantharaman, Yutong He, Zeynep Akata, Tommi Jaakkola, Nicholas Matthew Boffi, and Max Simchowitz. Diamond maps: Efficient reward alignment via stochastic flow maps. *arXiv preprint arXiv:2602.05993*, 2026.

[37] Peter Holderrieth, Uriel Singer, Tommi Jaakkola, Ricky TQ Chen, Yaron Lipman, and Brian Karrer. Glass flows: Transition sampling for alignment of flow and diffusion models. *arXiv preprint arXiv:2509.25170*, 2025.

[38] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

[39] Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv preprint arXiv:1907.00456*, 2019.

[40] Bingyi Kang, Xiao Ma, Chao Du, Tianyu Pang, and Shuicheng Yan. Efficient diffusion policies for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36:67195–67212, 2023.

[41] Hyeongyu Kang, Jaewoo Lee, Woocheol Shin, Kiyoung Om, and Jinkyoo Park. Diffusion fine-tuning via reparameterized policy gradient of the soft q-function. *arXiv preprint arXiv:2512.04559*, 2025.

[42] Shyamgopal Karthik, Karsten Roth, Massimiliano Mancini, and Zeynep Akata. If at first you don't succeed, try, try again: Faithful diffusion-based text-to-image generation by selection. *arXiv preprint arXiv:2305.13308*, 2023.

[43] Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. In *The Twelfth International Conference on Learning Representations*.

[44] Sunwoo Kim, Minkyu Kim, and Dongmin Park. Test-time alignment of diffusion models without reward over-optimization. In *The Thirteenth International Conference on Learning Representations*.

[45] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *stat*, 1050:1, 2014.

[46] Tomasz Korbak, Ethan Perez, and Christopher Buckley. Rl with kl penalties is better viewed as bayesian inference. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1083–1091, 2022.

[47] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*.

[48] Chieh-Hsin Lai, Bac Nguyen, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yuki Mitsufuji, Stefano Ermon, and Molei Tao. A unified view of drifting and score-based models. *arXiv preprint arXiv:2603.07514*, 2026.

[49] Jaewoo Lee, Minsu Kim, Sanghyeok Choi, Inhyuck Song, Sujin Yun, Hyeongyu Kang, Woocheol Shin, Taeyoung Yun, Kiyoung Om, and Jinkyoo Park. Diffusion alignment as variational expectation-maximization. *arXiv preprint arXiv:2510.00502*, 2025.

[50] Qiyang Li and Sergey Levine. Q-learning with adjoint matching. *arXiv preprint arXiv:2601.14234*, 2026.

[51] Qiyang Li, Seohong Park, and Sergey Levine. Decoupled q-chunking. *arXiv preprint arXiv:2512.10926*, 2025.

[52] Shufan Li, Konstantinos Kallidromitis, Akash Gokul, Yusuke Kato, and Kazuki Kozuka. Aligning diffusion models by optimizing human utility. *Advances in Neural Information Processing Systems*, 37:24897–24925, 2024.

[53] Xiner Li, Yulai Zhao, Chenyu Wang, Gabriele Scalia, Gökcen Eraslan, Surag Nair, Tommaso Biancalani, Shuiwang Ji, Aviv Regev, Sergey Levine, et al. Derivative-free guidance in continuous and discrete diffusion models with soft value-based decoding. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[54] Yingzhen Li and Richard E Turner. Gradient estimators for implicit models. In *International Conference on Learning Representations*, 2018.

[55] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*.

[56] 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*.

[57] Qiang Liu. Stein variational gradient descent as gradient flow. *Advances in neural information processing systems*, 30, 2017.

[58] Qiang Liu, Jason Lee, and Michael Jordan. A kernelized stein discrepancy for goodness-of-fit tests. In *International conference on machine learning*, pages 276–284. PMLR, 2016.

[59] Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. *Advances in neural information processing systems*, 29, 2016.

[60] 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 (ICLR)*, 2023.

[61] Zhen Liu, Tim Z Xiao, Carles Domingo-Enrich, Weiyang Liu, and Dinghuai Zhang. Value gradient guidance for flow matching alignment. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[62] Cheng Lu, Huayu Chen, Jianfei Chen, Hang Su, Chongxuan Li, and Jun Zhu. Contrastive energy prediction for exact energy-guided diffusion sampling in offline reinforcement learning. In *International Conference on Machine Learning*, pages 22825–22855. PMLR, 2023.

[63] Cheng Lu and Yang Song. Simplifying, stabilizing and scaling continuous-time consistency models. In *The Thirteenth International Conference on Learning Representations*.

[64] 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.

[65] Weijian Luo, Tianyang Hu, Shifeng Zhang, Jiacheng Sun, Zhenguo Li, and Zhihua Zhang. Diff-instruct: A universal approach for transferring knowledge from pre-trained diffusion models. *Advances in Neural Information Processing Systems*, 36:76525–76546, 2023.

[66] Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

[67] Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

[68] Yurii Nesterov and Vladimir Spokoiny. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17(2):527–566, 2017.

[69] Chris J Oates, Mark Girolami, and Nicolas Chopin. Control functionals for monte carlo integration. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 79(3):695–718, 2017.

[70] Owen Oertell, Jonathan D Chang, Yiyi Zhang, Kianté Brantley, and Wen Sun. Rl for consistency models: Faster reward guided text-to-image generation. *arXiv preprint arXiv:2404.03673*, 2024.

[71] Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *The Thirteenth International Conference on Learning Representations*.

[72] Seohong Park, Kevin Frans, Sergey Levine, and Aviral Kumar. Is value learning really the main bottleneck in offline rl? *Advances in Neural Information Processing Systems*, 37:79029–79056, 2024.

[73] Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. In *International Conference on Machine Learning (ICML)*, 2025.

[74] Emanuel Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076, 1962.

[75] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

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

[77] 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*, pages 8748–8763. PmLR, 2021.

[78] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

[79] Lorenz Richter, Ayman Boustati, Nikolas Nüsken, Francisco Ruiz, and Omer Deniz Akyildiz. Vargrad: a low-variance gradient estimator for variational inference. *Advances in Neural Information Processing Systems*, 33:13481–13492, 2020.

[80] 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.

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

[82] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *ACM SIGGRAPH 2022 conference proceedings*, pages 1–10, 2022.

[83] Christoph Schuhmann and Romain Beaumont. Laion-aesthetics, 2022. *URL https://laion.ai/blog/laion-aesthetics/. Accessed*, pages 11–10, 2023.

[84] David W Scott. Multivariate density estimation. 2015.

[85] Marta Skreta, Tara Akhound-Sadegh, Viktor Ohanesian, Roberto Bondesan, Alan Aspuru-Guzik, Arnaud Doucet, Rob Brekelmans, Alexander Tong, and Kirill Neklyudov. Feynman-kac correctors in diffusion: Annealing, guidance, and product of experts. In *International Conference on Machine Learning*, pages 55906–55949. PMLR, 2025.

[86] Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. In *The Twelfth International Conference on Learning Representations*.

[87] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, pages 32211–32252, 2023.

[88] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

[89] 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*.

[90] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147. pmlr, 2013.

[91] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

[92] Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36:11592–11620, 2023.

[93] 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.

[94] Masatoshi Uehara, Yulai Zhao, Kevin Black, Ehsan Hajiramezanali, Gabriele Scalia, Nathaniel Lee Diamant, Alex M Tseng, Tommaso Biancalani, and Sergey Levine. Fine-tuning of continuous-time diffusion models as entropy-regularized control. *arXiv preprint arXiv:2402.15194*, 2024.

[95] Masatoshi Uehara, Yulai Zhao, Chenyu Wang, Xiner Li, Aviv Regev, Sergey Levine, and Tommaso Biancalani. Inference-time alignment in diffusion models with reward-guided generation: Tutorial and review. *arXiv preprint arXiv:2501.09685*, 2025.

[96] Robert A. Vandermeulen and Clayton D. Scott. Consistency of robust kernel density estimators. In *Proceedings of the 26th Annual Conference on Learning Theory*, pages 568–591. PMLR, 2013.

[97] 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.

[98] Andrew Wagenmaker, Yunchu Zhang, Mitsuhiko Nakamoto, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. In *9th Annual Conference on Robot Learning*.

[99] Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8228–8238, 2024.

[100] Dilin Wang and Qiang Liu. Learning to draw samples: With application to amortized mle for generative adversarial learning. 2016.

[101] 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. *CoRR*, 2023.

[102] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[103] 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.

[104] 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*, 2025.

[105] 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.

[106] Yuchen Yang, Bo Hui, Haolin Yuan, Neil Gong, and Yinzhi Cao. Sneakyprompt: Jailbreaking text-to-image generative models. In *2024 IEEE symposium on security and privacy (SP)*, pages 897–912. IEEE, 2024.

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

[108] 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*, pages 6613–6623, 2024.

[109] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[110] Shiyuan Zhang, Weitong Zhang, and Quanquan Gu. Energy-weighted flow matching for offline reinforcement learning. In *The Thirteenth International Conference on Learning Representations*.

[111] XiangCheng Zhang, Haowei Lin, Haotian Ye, James Zou, Jianzhu Ma, Yitao Liang, and Yilun Du. Inference-time scaling of diffusion models through classical search. In *NeurIPS 2025 Workshop on Structured Probabilistic Inference {\&} Generative Modeling*.

[112] 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.

[113] Linqi Zhou, Stefano Ermon, and Jiaming Song. Inductive moment matching. In *Forty-second International Conference on Machine Learning*.

[114] Yuhao Zhou, Jiaxin Shi, and Jun Zhu. Nonparametric score estimators. In *International Conference on Machine Learning*, pages 11513–11522. PMLR, 2020.

[115] Huaisheng Zhu, Teng Xiao, and Vasant G Honavar. Dspo: Direct score preference optimization for diffusion model alignment. In *The Thirteenth International Conference on Learning Representations*.

Appendix

A Implementation details

In this section, we elaborate on implementation details for our experiments: toy setting, reinforcement learning, and image generator alignment.

A.1 Toy experiments

Dataset. All experiments are conducted on the 8-Gaussians dataset [62]. The data distribution $p(x)$ is an equally weighted mixture of eight isotropic Gaussians whose centers are uniformly placed on a circle of radius 4, with component standard deviation 0.5. All samples are then globally rescaled by $1/\sqrt{2}$, yielding effective centers on a circle of radius $4/\sqrt{2} \approx 2.83$ and an effective per-component standard deviation of $0.5/\sqrt{2} \approx 0.354$. For training, we draw a pool of 500,000 samples once per run. For evaluation, we additionally draw a fixed reference set of 10,000 samples and reuse it for all metric computations during the run.

Reward function. The fine-tuning target is the reward-tilted distribution

$$q^*(x) \propto p(x)\, \exp(r(x)),$$

with $\beta = 1$. On the 8-Gaussians dataset, we use a soft cluster-assignment reward defined as

$$\exp(r(x)) = \sum_{k=0}^{7} \mathrm{softmax}\big(-\|x - c_k\|^2\big)_k \cdot (k/7),$$

where $\{c_k\}_{k=0}^{7}$ are the eight cluster centers defining the data distribution. Thus, cluster k receives target reward $k/7$, making the eighth mode the most desirable and the first mode neutral. The softmax formulation makes the reward smooth and differentiable everywhere, although its gradient saturates near the cluster centers.

Pre-training. We pre-train VAE, Drifting model and MeanFlow on a fixed 2D toy dataset. All three models share a common backbone: 3-layer MLPs with hidden width 256 and SiLU activations [22]. They are optimized for 1M gradient steps with a batch size of 8192.

- **Variational Autoencoder [45].** The VAE consists of an encoder $q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \mathrm{diag}(\exp \sigma_\phi^2(x)))$ and a decoder $p_\theta(x|z)$, both parameterized as three-layer SiLU MLPs with 256 hidden units. The latent dimension is $d_z = 8$, and the prior is $p(z) = \mathcal{N}(0, I)$. We optimize the negative ELBO:

  $$\mathcal{L}_{\mathrm{VAE}}(x) = \frac{1}{2}\|x - \hat{x}\|_2^2 + \beta\, \mathrm{KL}(q_\phi(z \mid x) \,\|\, \mathcal{N}(0, I)) \,.$$

  We clamp logvar to $[-8, 8]$ for numerical stability and use the reparameterization $z = \mu + \exp(\frac{1}{2}\sigma^2)\epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$. We set $\beta = 0.02$.

- **Drifting model [17].** We instantiate the drifting model as a one-step pushforward map $f_\theta : \mathbb{R}^{32} \to \mathbb{R}^2$ with latent noise $\epsilon \sim \mathcal{N}(0, I_{32})$, parameterized by a three-layer SiLU MLP. Let $p_\theta = (f_\theta)_{\#} p_\epsilon$ denote the generated distribution. Given positive samples $y^+ \sim p_{\mathrm{data}}$ and negative samples $y^- \sim p_\theta$, the drifting field is defined as

  $$V_{p_{\mathrm{data}}, p_\theta}(x) = \frac{1}{Z_{p_{\mathrm{data}}}(x) Z_{p_\theta}(x)} \mathbb{E}_{y^+ \sim p_{\mathrm{data}},\, y^- \sim p_\theta} \left[k(x, y^+)k(x, y^-)(y^+ - y^-)\right],$$

  where $Z_{p_{\mathrm{data}}}(x) := \mathbb{E}_{y^+ \sim p_{\mathrm{data}}} \left[k(x, y^+)\right]$, $Z_{p_\theta}(x) := \mathbb{E}_{y^- \sim p_\theta} \left[k(x, y^-)\right]$. We use the kernel $k(x, y) = \exp(-\|x - y\|_2/\tau)$. The generator is trained by regression to the stop-gradient drifted target:

  $$\mathcal{L}_{\mathrm{drift}} = \|\mathrm{stopgrad}(f_\theta(\epsilon) + V_{p_{\mathrm{data}}, p_\theta}(f_\theta(\epsilon)) - f_\theta(\epsilon))\|_2^2 \,.$$

  We set the drifting temperature to $\tau = 0.15$.

- **MeanFlow [29].** The MeanFlow model learns the mean velocity $u_\theta(x_t, r, t)$ of a linear interpolation coupling $x_t = (1 - t)x_0 + tx_1$ between data $x_0 \sim p_{\mathrm{data}}$ at $t = 0$ and noise $x_1 \sim \mathcal{N}(0, I)$ at $t = 1$. It is instantiated as a three-layer SiLU MLP taking $[x_t, t, t - r]$ as input. The pairs (t, r) are drawn as $(\max, \min)$ of two $\mathrm{sigmoid}(\mathcal{N}(-0.4, 1))$ samples, with probability $p_{\mathrm{eq}} = 0.5$ replaced by $r = t$. Let $v_t = x_1 - x_0$ denote the instantaneous velocity. The Jacobian-Vector Product (JVP) $\partial u_\theta / \partial t$ along $(v_t, 0, 1)$ gives the MeanFlow target

  $$u^\star(x_t, r, t) = \mathrm{stopgrad}\left(v_t - (t - r)\frac{du_\theta}{dt}\right).$$

Table 5: **Hyperparameters for toy experiments.**

Component	VAE	Drifting model	MeanFlow
Optimizer	Adam	Adam	AdamW
(β_1, β_2)	$(0.9, 0.999)$	$(0.9, 0.999)$	$(0.9, 0.95)$
Weight decay	0	0	0
Learning rate	1×10^{-3}	1×10^{-3}	3×10^{-4}
Batch size	8192	8192	8192
Training steps	1×10^6	1×10^6	1×10^6
Hidden width / depth	256 / 3	256 / 3	256 / 3
Activation	SiLU	SiLU	SiLU
Latent / input dim	$d_z = 8$	$d_z = 32$	—
Gradient clipping	—	—	$\|g\|_2 \leq 1$

The objective is the adaptively weighted MSE:

$$\mathcal{L}_{\text{MF}} = \frac{\|u_\theta(x_t, r, t) - u^\star\|_2^2}{(\text{stopgrad}(\|u_\theta - u^\star\|_2^2) + 10^{-2})^p},$$

with adaptive-weighting power $p = 1.0$. Gradients are clipped to $\|\cdot\|_2 \leq 1$.

Fine-tuning baselines. For MeanFlow fine-tuning with Regularized REINFORCE and Adjoint Matching, we approximate the MeanFlow mapping $u_\theta(x_t, r, t)$ as an instantaneous velocity v_t, allowing us to formulate the sampling as an SDE. We inject Gaussian noise where the noise scale follows a VP-SDE-style [89] schedule $dx = -v(x,t)\,dt + \sigma(t)\,dW$, $\sigma(t) = \eta\sqrt{t/(1-t)}$ for the regularized REINFORCE and memoryless schedule $\sigma(t) = \sqrt{2t/(1-t)}$ for the Adjoint Matching. Since applying Regularized REINFORCE and Adjoint Matching to one-step noise-to-data mapping is non-trivial, we evaluate the fine-tuning baselines only on the multi-step MeanFlow backbone. We fine-tune each algorithm for 20,000 gradient steps with a 1024 batch size and select the best checkpoint according to the lowest exponentially averaged KLD.

- **Regularized REINFORCE.** For Regularized REINFORCE, we implement the on-policy variant of Relative Trajectory Balance (RTB) [97], which is equivalent to KL-regularized REINFORCE [15]. We use VarGrad [79] to estimate the normalizing constant of the RTB objective. The noise scale η is selected by grid search over $\{0.01, 0.05, 0.1, 0.2, 0.3, 0.5\}$. The selected values are $\eta = 0.01$ for $n = 2$, $\eta = 0.05$ for $n = 4$, $\eta = 0.10$ for $n = 8$, and $\eta = 0.20$ for $n = 16$, where n denotes the number of iterative denoising steps.
- **Adjoint Matching.** In the case of Adjoint Matching, we follow the implementation guideline provided in the original paper [21]. Specifically, we implement the memoryless noise schedule $\sigma(t) = \sqrt{2(t+h)/(1-t+h)}$, where h denotes the time discretization step size, for numerical stability and faster fine-tuning. For gradient evaluation, we subsample half of the timesteps: we uniformly sample timesteps from the first 75% of the trajectory and always include all timesteps in the final 25%. (See details in Appendix G in [21].)

A.2 Offline and Offline-to-Online RL

We implement FAV on the codebases of FQL [1] and QAM [2] using the JAX [8] implementation. We greatly thank the authors of [73] for providing highly reproducible codebases. Please note that FAV shares the same architecture for the actor and critic, latent parameterization, optimizer hyperparameters, critic settings, and normalization technique, as well as the offline-to-online adaptation protocol, with FQL [73].

Benchmarks and training protocol. We evaluate FAV on 56 offline RL tasks from OGBench and D4RL AntMaze. From OGBench, we use the reward-based single-task variants, covering 50 tasks across 10 environments, and additionally include 6 challenging AntMaze tasks from D4RL. For offline-to-online RL, we select the six OGBench environments on which FAV attains the lowest offline RL performance, corresponding to 30 tasks in total. Following the prior experimental evaluation protocol [73], agents are trained for 1M gradient steps in the offline setting, and for offline-to-online RL we continue training for an additional 1M online steps. For D4RL AntMaze, agents are trained for 500K steps. We report results over 8 random seeds as the mean with 95% confidence intervals.

[1] https://github.com/seohongpark/fql
[2] https://github.com/ColinQiyangLi/qam

Policy and critic architectures. We instantiate FAV as a single-step noise-to-action policy that maps the concatenation of the observation and latent noise directly to an action. The actor is implemented as a multilayer perceptron with 4 hidden layers of width 512, GELU activations, and no normalization layers. The critic is a double-Q ensemble with two parallel value heads, each using 4 hidden layers of width 512 and GELU activations, with LayerNorm applied after each activation. Both actor and critic use variance-scaling initialization with fan-average uniform kernels. The critic output layer is linear without a final activation, while the actor output is linear and clipped to the valid action range.

Latent parameterization and optimization. For action generation, we sample latent noise as $z \in \mathbb{R}^a \sim \mathcal{N}(0, I)$ and feed the concatenated vector $[s; z] \in \mathbb{R}^{s+a}$ to the actor, where s and a denote the dimension of state and action, respectively. Both the actor and the critic are optimized using Adam with a learning rate of 3×10^{-4} and a batch size of 256. A target critic is updated at every gradient step using Polyak averaging with coefficient $\tau = 0.005$.

Critic training and normalization. We use a double-Q critic and compute the Bellman target using the mean of the target ensemble by default, although we also support a minimum aggregation variant. The target is given by

$$y = r + \gamma(1 - d)\,\mathrm{mean}\big(Q_{\mathrm{target}}(s', a')\big),$$

where $a' \sim \pi(s')$, and d denotes the termination flag. The critic is trained with a mean-squared Bellman error. We use a discount factor $\gamma = 0.99$ by default and set $\gamma = 0.995$ for `antmaze-giant`, `humanoidmaze`, and `antsoccer` environments. We do not normalize observations, and actions are clipped to $[-1, 1]$ during generation. Rewards are left unscaled except for D4RL AntMaze, where we apply the standard shift $r \leftarrow r - 1$.

Offline-to-Online fine-tuning. For online adaptation, FAV uses the empirical action distribution induced by the replay buffer as the reference distribution and preserves the same core objective used in offline training. We use a single circular replay buffer initialized with the offline dataset and sample uniformly from it thereafter. The replay buffer capacity is 2M transitions, and we use an update-to-data ratio of 1, i.e., one gradient step per environment interaction. During online fine-tuning, exploration is induced solely through the stochastic one-step policy by sampling fresh Gaussian noise at each interaction step, without any additional exploration bonus or ϵ-greedy strategy.

FAV hyperparameters. FAV introduces β that controls the strength of value-guided alignment, along with a kernel bandwidth τ. In our implementation, we sweep β over {0.5, 1, 2, 3, 5}. We search the kernel bandwidth τ from {0.05, 0.1, 0.5, 1.0}. For FAV-Adaptive, we determine the kernel bandwidth directly from data using Scott's rule.

Kernel specification. FAV uses a Gaussian RBF kernel,

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{\tau}\right), \qquad \tau = 2\sigma^2, \tag{9}$$

to construct the Stein velocity field. We employ equal bandwidth for both of two kernels. For each state, we generate $N = 8$ action particles and compute the drift using three components: a prior-score term, a value-gradient term, and a repulsive term between generated particles.

Evaluation protocol. We report results over 8 random seeds as mean $\pm$ 95% confidence interval. For OGBench, agents are trained for 1M offline steps, and in the offline-to-online setting, we continue training for an additional 1M online steps. For D4RL antmaze, agents are trained for 500K steps. Following the protocol of Park et al. [73], we report offline OGBench performance averaged over 800K, 900K, and 1M steps, and D4RL antmaze performance averaged over checkpoints at 300K, 400K, and 500K steps. For offline-to-online RL, we report performance at 1M steps and 2M steps. The results of baselines are carried over from prior works [98], while we reproduce the QAM with the same critic and actor architecture and hyperparameters shared with FQL and FAV.

Baselines for Offline RL tasks. The results in Table 1 are drawn from multiple sources. Results for IQL, ReBRAC, FAWAC, IFQL, SRPO, and CAC are taken from Park et al. [73], which conducted an extensive hyperparameter search. DSRL results on OGBench are taken from Wagenmaker et al. [98]. Since the original DSRL paper [98] does not report results on the D4RL antmaze tasks, we reproduce them using the QAM codebase [50]. For consistency with our benchmark, we replace the 10-head Q-ensemble and Q-chunking [51] on the implementation of DSRL on the QAM codebase with a 2-head Q-function without Q-chunking. Following the guidelines of Li et al. [50], we grid-search σ_z over {0.1,0.2,0.4,0.6,0.8,1,1.2,1.4}. We also reproduce QAM as a recent baseline. For a fair comparison, we again adopt a 2-head Q-function without Q-chunking, keeping the same policy and critic architectures as in the other baselines and FAV. Following [50], we grid-search τ over {0.1, 0.3, 1, 3, 10, 30}.

Table 6: **RL tasks hyperparameters** Methods unavailable in the source are left as '–'.

Task	Gaussian Policies		Latent Opt.	Flow-based Policies			Distillation Policies			Ours	
	IQL	ReBRAC	DSRL	FAWAC	IFQL	QAM	SRPO	CAC	FQL	FAV-Adaptive	FAV
	α	(α_1, α_2)	σ_z	α	N	τ	β	η	α	β	(τ, β)
antmaze-large-navigate-singletask-task1-v0	10	(0.003, 0.01)	1.25	3	32	3	0.3	1	10	1	(0.5, 2)
antmaze-large-navigate-singletask-task2-v0	10	(0.003, 0.01)	1.25	3	32	3	0.3	1	10	1	(0.5, 2)
antmaze-large-navigate-singletask-task3-v0	10	(0.003, 0.01)	1.25	3	32	3	0.3	1	10	1	(0.5, 2)
antmaze-large-navigate-singletask-task4-v0	10	(0.003, 0.01)	1.25	3	32	3	0.3	1	10	1	(0.5, 2)
antmaze-large-navigate-singletask-task5-v0	10	(0.003, 0.01)	1.25	3	32	3	0.3	1	10	1	(0.5, 2)
antmaze-giant-navigate-singletask-task1-v0	10	(0.003, 0.01)	1.25	3	32	0.3	0.3	1	10	1	(0.5, 1)
antmaze-giant-navigate-singletask-task2-v0	10	(0.003, 0.01)	1.25	3	32	0.3	0.3	1	10	1	(0.5, 1)
antmaze-giant-navigate-singletask-task3-v0	10	(0.003, 0.01)	1.25	3	32	0.3	0.3	1	10	1	(0.5, 1)
antmaze-giant-navigate-singletask-task4-v0	10	(0.003, 0.01)	1.25	3	32	0.3	0.3	1	10	1	(0.5, 1)
antmaze-giant-navigate-singletask-task5-v0	10	(0.003, 0.01)	1.25	3	32	0.3	0.3	1	10	1	(0.5, 1)
humanoidmaze-medium-navigate-singletask-task1-v0	10	(0.01, 0.01)	0.5	3	32	3	0.3	0.03	30	2	(0.5, 1)
humanoidmaze-medium-navigate-singletask-task2-v0	10	(0.01, 0.01)	0.5	3	32	3	0.3	0.03	30	2	(0.5, 1)
humanoidmaze-medium-navigate-singletask-task3-v0	10	(0.01, 0.01)	0.5	3	32	3	0.3	0.03	30	2	(0.5, 1)
humanoidmaze-medium-navigate-singletask-task4-v0	10	(0.01, 0.01)	0.5	3	32	3	0.3	0.03	30	2	(0.5, 1)
humanoidmaze-medium-navigate-singletask-task5-v0	10	(0.01, 0.01)	0.5	3	32	3	0.3	0.03	30	2	(0.5, 1)
humanoidmaze-large-navigate-singletask-task1-v0	10	(0.01, 0.01)	0.75	3	32	3	0.3	1	30	2	(0.5, 1)
humanoidmaze-large-navigate-singletask-task2-v0	10	(0.01, 0.01)	0.75	3	32	3	0.3	1	30	2	(0.5, 1)
humanoidmaze-large-navigate-singletask-task3-v0	10	(0.01, 0.01)	0.75	3	32	3	0.3	1	30	2	(0.5, 1)
humanoidmaze-large-navigate-singletask-task4-v0	10	(0.01, 0.01)	0.75	3	32	3	0.3	1	30	2	(0.5, 1)
humanoidmaze-large-navigate-singletask-task5-v0	10	(0.01, 0.01)	0.75	3	32	3	0.3	1	30	2	(0.5, 1)
antsoccer-arena-navigate-singletask-task1-v0	1	(0.01, 0.01)	0.75	10	64	3	0.03	1	10	2	(1, 1)
antsoccer-arena-navigate-singletask-task2-v0	1	(0.01, 0.01)	0.75	10	64	3	0.03	1	10	2	(1, 1)
antsoccer-arena-navigate-singletask-task3-v0	1	(0.01, 0.01)	0.75	10	64	3	0.03	1	10	2	(1, 1)
antsoccer-arena-navigate-singletask-task4-v0	1	(0.01, 0.01)	0.75	10	64	3	0.03	1	10	2	(1, 1)
antsoccer-arena-navigate-singletask-task5-v0	1	(0.01, 0.01)	0.75	10	64	3	0.03	1	10	2	(1, 1)
cube-single-play-singletask-task1-v0	1	(1, 0)	0.5	1	32	3	0.03	0.003	300	0.5	(0.05, 1)
cube-single-play-singletask-task2-v0	1	(1, 0)	0.5	1	32	3	0.03	0.003	300	0.5	(0.05, 1)
cube-single-play-singletask-task3-v0	1	(1, 0)	0.5	1	32	3	0.03	0.003	300	0.5	(0.05, 1)
cube-single-play-singletask-task4-v0	1	(1, 0)	0.5	1	32	3	0.03	0.003	300	0.5	(0.05, 1)
cube-single-play-singletask-task5-v0	1	(1, 0)	0.5	1	32	3	0.03	0.003	300	0.5	(0.05, 1)
cube-double-play-singletask-task1-v0	0.3	(0.1, 0)	1.5	0.3	32	0.3	0.1	0.3	300	0.5	(0.1, 0.5)
cube-double-play-singletask-task2-v0	0.3	(0.1, 0)	1.5	0.3	32	0.3	0.1	0.3	300	0.5	(0.1, 0.5)
cube-double-play-singletask-task3-v0	0.3	(0.1, 0)	1.5	0.3	32	0.3	0.1	0.3	300	0.5	(0.1, 0.5)
cube-double-play-singletask-task4-v0	0.3	(0.1, 0)	1.5	0.3	32	0.3	0.1	0.3	300	0.5	(0.1, 0.5)
cube-double-play-singletask-task5-v0	0.3	(0.1, 0)	1.5	0.3	32	0.3	0.1	0.3	300	0.5	(0.1, 0.5)
scene-play-singletask-task1-v0	10	(0.1, 0.01)	0.75	0.3	32	0.3	0.1	0.3	300	0.5	(0.1, 0.5)
scene-play-singletask-task2-v0	10	(0.1, 0.01)	0.75	0.3	32	0.3	0.1	0.3	300	0.5	(0.1, 0.5)
scene-play-singletask-task3-v0	10	(0.1, 0.01)	0.75	0.3	32	0.3	0.1	0.3	300	0.5	(0.1, 0.5)
scene-play-singletask-task4-v0	10	(0.1, 0.01)	0.75	0.3	32	0.3	0.1	0.3	300	0.5	(0.1, 0.5)
scene-play-singletask-task5-v0	10	(0.1, 0.01)	0.75	0.3	32	0.3	0.1	0.3	300	0.5	(0.1, 0.5)
puzzle-3x3-play-singletask-task1-v0	10	(0.3, 0.01)	0.5	0.3	32	0.1	0.1	0.01	1000	2	(0.5, 0.5)
puzzle-3x3-play-singletask-task2-v0	10	(0.3, 0.01)	0.5	0.3	32	0.1	0.1	0.01	1000	2	(0.5, 0.5)
puzzle-3x3-play-singletask-task3-v0	10	(0.3, 0.01)	0.5	0.3	32	0.1	0.1	0.01	1000	2	(0.5, 0.5)
puzzle-3x3-play-singletask-task4-v0	10	(0.3, 0.01)	0.5	0.3	32	0.1	0.1	0.01	1000	2	(0.5, 0.5)
puzzle-3x3-play-singletask-task5-v0	10	(0.3, 0.01)	0.5	0.3	32	0.1	0.1	0.01	1000	2	(0.5, 0.5)
puzzle-4x4-play-singletask-task1-v0	3	(0.3, 0.01)	0.5	0.3	32	0.1	0.1	0.01	1000	0.1	(1, 0.5)
puzzle-4x4-play-singletask-task2-v0	3	(0.3, 0.01)	0.5	0.3	32	0.1	0.1	0.01	1000	0.1	(1, 0.5)
puzzle-4x4-play-singletask-task3-v0	3	(0.3, 0.01)	0.5	0.3	32	0.1	0.1	0.01	1000	0.1	(1, 0.5)
puzzle-4x4-play-singletask-task4-v0	3	(0.3, 0.01)	0.5	0.3	32	0.1	0.1	0.01	1000	0.1	(1, 0.5)
puzzle-4x4-play-singletask-task5-v0	3	(0.3, 0.01)	0.5	0.3	32	0.1	0.1	0.01	1000	0.1	(1, 0.5)
antmaze-umaze-v2	–	–	1.4	3	32	3	–	0.01	10	3	(0.5, 1)
antmaze-umaze-diverse-v2	–	–	1.2	3	32	3	–	0.01	10	3	(1, 1)
antmaze-medium-play-v2	–	–	1.4	3	32	3	–	0.01	10	2	(1, 3)
antmaze-medium-diverse-v2	–	–	1.0	3	32	3	–	0.01	10	3	(1, 3)
antmaze-large-play-v2	–	–	1.4	3	32	10	–	4.5	3	5	(1, 3)
antmaze-large-diverse-v2	–	–	1.2	3	32	10	–	3.5	3	5	(1, 3)

Baselines for Offline-to-Online RL tasks. For the offline-to-online setting, we reproduce the full benchmark, including IQL, ReBRAC, RLPD, IFQL, and FQL—to obtain performance curves over training steps. Hyperparameters are shared between the offline and offline-to-online phases. All experiments are built on the official codebase of FQL.

Automated bandwidth for FAV-Adaptive FAV-Adaptive sets the temperature from the dataset statistics. We use Scott's rule with the action scale computed once from the full offline dataset $\mathcal{D}$. Let d_a be the action dimension and $\hat{\sigma}_{\mathcal{D}} = \frac{1}{d_a} \sum_{j=1}^{d_a} \mathrm{Std}_{a \sim \mathcal{D}}(a_j)$. Given batch size n, we set $h_{\mathrm{Scott}} = n^{-1/(d_a+4)} \hat{\sigma}_{\mathcal{D}}$, $\tau_{\mathrm{adaptive}} = 2h^2_{\mathrm{Scott}}$. FAV-Adaptive uses the full-dataset action statistics to determine a fixed environment-specific kernel temperature, whereas the sample-size factor scales with the training batch size.

A.3 Conditional image generation

Model specification The following table lists the official implementations of the pre-trained models used in our experiments. For ImageNet experiments, we use the ImageNet 256×256 pre-trained checkpoints provided in each repository.

Models	Links
StyleGAN-XL [82]	`https://github.com/autonomousvision/stylegan-xl`
Drifting Model-L/latent [17]	`https://github.com/lambertae/drifting`
IMM-XL/2 [113]	`https://github.com/lumalabs/imm`
iMeanFlow-XL/2 [30]	`https://github.com/Lyy-iiis/imeanflow/tree/torch`
Sana-Sprint 1.6B [10]	`https://github.com/NVlabs/Sana`

Pre-trained encoder We instantiate the pre-trained encoder ψ in Equation (8) with the vision backbone of each reward model, ensuring that kernel proximity reflects reward-relevant semantic similarity of samples. Specifically: (i) OpenAI CLIP ViT-L/14 for the LAION aesthetic reward [83]; (ii) OpenCLIP ViT-H/14 (LAION-2B pre-trained) fine-tuned on Human Preference Dataset v2 for HPS [101]; (iii) ViT-base (ImageNet-21k pre-trained) fine-tuned on ∼80K normal/NSFW images by Falconsai for NSFW [24].

Batch for training For each of N_c conditions, we draw N_{gen} samples from the current model q_θ and N_{ref} samples from the pre-trained model p_{ref}. We compute the loss over these $N_{\text{gen}} + N_{\text{ref}}$ samples per condition and aggregate across N_c conditions, yielding an effective batch size of $N_c \cdot (N_{\text{gen}} + N_{\text{ref}})$. In all experiments, we set $N_{\text{gen}} = N_{\text{ref}} = 16$ and $N_c = 8$, resulting in an effective batch size of 256. All baseline methods are trained with the same effective batch size.

Hyperparameters for training Table 7 summarizes the training hyperparameters used in each experimental setting. When sampling from the pre-trained models, we follow the default hyperparameters (e.g. CFG scale) provided in each official repository.

Table 7: Training hyperparameters for each experimental setting.

	ImageNet-Aesthetic	Sana-NSFW	Sana-HPS
Learning rate	0.0005	0.0005	0.0005
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.95$)	AdamW ($\beta_1 = 0.9, \beta_2 = 0.95$)	AdamW ($\beta_1 = 0.9, \beta_2 = 0.95$)
Weight decay	0.01	0.01	0.01
Gradient clip norm	2	2	1
Max steps	200	100	200
Batch sizes	256	256	256

Hyperparameters for FAV Two key hyperparameters in FAV are the alignment coefficient β, which controls the strength of reward guidance, and the Gaussian RBF kernel bandwidth τ in Equation (9), which determines the locality of SVGD interactions. We search over $\beta = \{0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100\}$ and $\tau = \{0.01, 0.1, 0.3, 0.5, 1\}$. We find that $\tau = 0.3$ empirically works well across all settings, and therefore fix it throughout our experiments. For LoRA fine-tuning [38], we select the rank and scale parameters based on the model backbone and the target reward. Table 8 reports the selected values for each experiment.

Table 8: Hyperparameters for FAV in conditional image generation

	Aesthetic				Compressibility	Incompressibility	NSFW	HPS
Backbone	StyleGAN	Drifting	IMM	iMeanFlow	iMeanFlow	iMeanFlow	Sana-sprint	Sana-sprint
β	10	1	0.5	0.5	0.01	0.05	0.1	100
τ	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3
LoRA rank	32	32	16	16	16	16	8	32
LoRA scale	1	4	4	4	4	4	1	1

Baselines for conditional image generation tasks We compare FAV against two families of alignment methods: fine-tuning approaches (DRaFT, Flow-GRPO+KL, RLCM, and Adjoint Matching) and test-time search approaches (Best-of-N and ReNO).

- **DRaFT** [14]: When reward overoptimization occurs too rapidly, we multiply the reward by a scaling factor to ensure a fair comparison with other baselines.
- **Flow-GRPO+KL** [56]: To approximate the sampling procedure of the flow-map model as ODE solving, we treat iMeanFlow's average velocity field as the instantaneous velocity field follwing Appendix A.1. We search over the noise level $\eta = \{0.01, 0.05, 0.1, 0.2\}$ and $\beta = \{1e-6, 1e-5, 1e-4, 5e-4, 1e-3, 1e-2, 0.5, 1.0\}$. We set the group size as 32.
- **RLCM** [70]: We adopt the RLCM MDP, replacing the consistency model [87] based policy with its sCM [63] counterpart, since our pre-trained model Sana-Sprint [10] is built on the sCM framework. All other components follow the official implementation.

- **Adjoint Matching** [21]: We fine-tune the flow-map model using the same approximation as Flow-GRPO+KL. We adopt the practical memoryless noise schedule, timestep selection to prioritize the last 25% of the sampling trajectory, and apply loss clipping via threshold following Appendix G in [21]. We search over LCT scalar $= \{1.6, 16, 160\}$ and $\beta = \{1, 5, 10, 50, 100, 500, 1000\}$.
- **Best-of-N**: We generate $N = 256$ samples per prompt from the pre-trained model and select the one with the highest reward.
- **ReNO** [23]: We follow the official implementation with 50 search steps, a learning rate of 5.0, and employ the Nesterov momentum optimizer [67, 90] with a regularizer coefficient of 0.01.

Prompts We report the class labels and prompts used in our experiments. For ImageNet256 experiments with aesthetic score as the target reward, we use 32 randomly sampled ImageNet class labels for both training and evaluation, motivated by the "simple animals" setting commonly used in diffusion alignment [4, 14]. The class labels reported in Table 9.

For high-resolution text-to-image experiments, we consider two target rewards. When optimizing HPS, we construct an evaluation set by sampling 32 prompts from the DrawBench prompts [80], with the number of prompts sampled from each category proportional to its category size. The sampled DrawBench prompts are listed in Table 10. When optimizing the NSFW classifier, we use SneakyPrompt [106]. We do not disclose the exact prompts, as they are adversarially constructed to elicit unsafe generations. We randomly sample 32 prompts for evaluation. In both tasks, training and evaluation prompts are disjoint.

Table 9: **Class labels used for ImageNet256 experiments.**

ID	Class	ID	Class	ID	Class	ID	Class
0	tench	9	ostrich	22	bald eagle	39	common iguana
55	green snake	69	trilobite	80	black grouse	105	koala
108	sea anemone	115	sea slug	130	flamingo	207	golden retriever
291	lion	387	lesser panda	398	abacus	403	aircraft carrier
404	airliner	409	analog clock	414	backpack	483	castle
497	church	540	drilling platform	547	electric locomotive	550	espresso maker
561	forklift	562	fountain	620	laptop	649	megalith
650	microphone	671	mountain bike	732	Polaroid camera	985	daisy

Table 10: **DrawBench prompts used for high-resolution text-to-image generation.**

#	Prompt
1	A blue bird and a brown bear.
2	A pink colored giraffe.
3	A white colored sandwich.
4	A yellow book and a red vase.
5	Rainbow coloured penguin.
6	An elephant under the sea.
7	Two cats and two dogs sitting on the grass.
8	Two cats and one dog sitting on the grass.
9	Two cats and three dogs sitting on the grass.
10	A triangular purple flower pot. A purple flower pot in the shape of a triangle.
11	A side view of an owl sitting in a field.
12	A cube made of brick. A cube with the texture of brick.
13	An instrument used for cutting cloth, paper, and other thin material, consisting of two blades laid one on top of the other and fastened in the middle so as to allow them to be opened and closed by a thumb and finger inserted through rings on the end of their handles.
14	An organ of soft nervous tissue contained in the skull of vertebrates, functioning as the coordinating center of sensation and intellectual and nervous activity.
15	A long curved fruit which grows in clusters and has soft pulpy flesh and yellow skin when ripe.
16	Paying for a quarter-sized pizza with a pizza-sized quarter.
17	A donkey and an octopus are playing a game. The donkey is holding a rope on one end, the octopus is holding onto the other. The donkey holds the rope in its mouth. A cat is jumping over the rope.
18	Tcennis rpacket.
19	Pafrking metr.
20	A giraffe underneath a microwave.
21	A banana on the left of an apple.
22	A carrot on the left of a broccoli.

#	Prompt
23	Jentacular.
24	Painting of the orange cat Otto von Garfield, Count of Bismarck-Schönhausen, Duke of Lauenburg, Minister-President of Prussia. Depicted wearing a Prussian Pickelhaube and eating his favorite meal—lasagna.
25	Illustration of a mouse using a mushroom as an umbrella.
26	Greek statue of a man tripping over a cat.
27	Darth Vader playing with raccoon in Mars during sunset.
28	McDonalds Church.
29	A realistic photo of a Pomeranian dressed up like a 1980s professional wrestler with neon green and neon orange face paint and bright green wrestling tights with bright orange boots.
30	A storefront with “Diffusion” written on it.
31	A storefront with “Google Research Pizza Cafe” written on it.
32	A storefront with “Hello World” written on it.

B KDE-based score estimation

We write the KDE-based score approximation in population form to emphasize the resulting Stein velocity field in Section 5.2. In practice, the expectation over p_{ref} is replaced by an empirical average over finite reference samples. Accordingly, this section derives the KDE-based score estimator in its finite-sample form.

B.1 Derivation of KDE-based score estimator

Let p be a density on $\mathbb{R}^d$ and let k_σ denote the Gaussian kernel with bandwidth $\sigma > 0$:

$$k_\sigma(u) = (2\pi\sigma^2)^{-d/2} \exp\left(-\frac{\|u\|^2}{2\sigma^2}\right). \tag{10}$$

Convolving p with k_σ gives the smoothed density:

$$p_\sigma(x) = (k_\sigma * p)(x) = \int k_\sigma(x-u)\, p(u)\, du. \tag{11}$$

Given i.i.d. samples $\{X_i\}_{i=1}^N$ from p, the **kernel density estimator** (KDE) approximates p_σ by replacing the expectation with an empirical average:

$$\hat{p}_\sigma(x) = \frac{1}{N}\sum_{i=1}^N k_\sigma(x - X_i), \tag{12}$$

so that $p_\sigma(x) = \mathbb{E}[\hat{p}_\sigma(x)]$. We now derive the KDE-based score estimator for $\nabla_x \log p(x)$, the score of the true density p. For the Gaussian kernel, differentiation gives:

$$\nabla_x k_\sigma(x-y) = \frac{y-x}{\sigma^2}\, k_\sigma(x-y). \tag{13}$$

Differentiating $\hat{p}_\sigma$ and dividing by $\hat{p}_\sigma(x)$ yields the score estimate:

$$\nabla_x \log \hat{p}_\sigma(x) = \frac{\nabla_x \hat{p}_\sigma(x)}{\hat{p}_\sigma(x)} = \sum_{i=1}^N \frac{k_\sigma(x-X_i)}{\sum_j k_\sigma(x-X_j)} \frac{X_i - x}{\sigma^2} = \sum_{i=1}^N \hat{\tilde{k}}_\sigma(x, X_i)\frac{X_i - x}{\sigma^2}, \tag{14}$$

which corresponds to the classical mean-shift vector [11]. Here $\hat{\tilde{k}}_\sigma$ are normalized weights summing to 1, making this a weighted average of directions $(X_i - x)$ with nearby points weighted more heavily.

In the FAV setting, the density of interest is the reference distribution p_{ref}, and the score is evaluated at a model sample $x' \sim q_\theta$. Thus, using reference samples $x_i^{\mathrm{ref}} \sim p_{\mathrm{ref}}$, the reference score can be approximated as

$$\nabla_{x'} \log p_{\mathrm{ref}}(x') \approx \sum_{i=1}^N \hat{\tilde{k}}_\sigma(x', x_i^{\mathrm{ref}})\frac{x_i^{\mathrm{ref}} - x'}{\sigma^2}, \tag{15}$$

where the empirical normalized weight is defined as

$$\hat{\tilde{k}}_\sigma(x', x_i^{\mathrm{ref}}) := \frac{k_\sigma(x' - x_i^{\mathrm{ref}})}{\sum_{j=1}^N k_\sigma(x' - x_j^{\mathrm{ref}})}. \tag{16}$$

Equivalently, in expectation notation, this becomes

$$\nabla_{x'} \log p_{\mathrm{ref}}(x') \approx \mathbb{E}_{x^{\mathrm{ref}} \sim p_{\mathrm{ref}}}\left[\tilde{k}_\sigma(x', x^{\mathrm{ref}})\frac{x^{\mathrm{ref}} - x'}{\sigma^2}\right], \tag{17}$$

with the population-level normalized weight

$$\tilde{k}_\sigma(x', x^{\mathrm{ref}}) := \frac{k_\sigma(x' - x^{\mathrm{ref}})}{\mathbb{E}_{\bar{x}^{\mathrm{ref}} \sim p_{\mathrm{ref}}}[k_\sigma(x' - \bar{x}^{\mathrm{ref}})]}. \tag{18}$$

The empirical weight $\hat{\tilde{k}}_\sigma$ is the finite-sample estimator of the population-level weight $\tilde{k}_\sigma$, and converges to $\tilde{k}_\sigma$ as $N \to \infty$. Substituting KDE-based approximation in Equation (17) into the prior-alignment term of Equation (5) recovers the form used in Equation (6).

B.2 Consistency condition of KDE-based surrogate target distribution

Recall that the true target distribution is $q^*(x) \propto p_{\mathrm{ref}}(x)\exp(\beta r(x))$ and KDE-based surrogate target distribution is $\hat{q}_\sigma^*(x) \propto \hat{p}_\sigma(x)\exp(\beta r(x))$. In this section, we establish that the $\hat{q}_\sigma^*$ consistently recovers the q^* under standard KDE regularity conditions.

For notational simplicity, we write p for p_{ref} and define

$$w(x) := \exp(\beta r(x)). \tag{19}$$

Then the true and surrogate target distributions are given by

$$q^*(x) = \frac{p(x)w(x)}{Z}, \qquad Z := \int p(x)w(x)\,dx, \tag{20}$$

and

$$\hat{q}^*_\sigma(x) = \frac{\hat{p}_\sigma(x)w(x)}{\hat{Z}_\sigma}, \qquad \hat{Z}_\sigma := \int \hat{p}_\sigma(x)w(x)\,dx. \tag{21}$$

We assume that (i) p is a valid density on $\mathbb{R}^d$, (ii) the kernel K is sufficiently smooth with finite moments; in particular, the Gaussian kernel k_σ satisfies these conditions, (iii) the reward weight is bounded, i.e., $\|w\|_\infty < \infty$, and (iv) the normalizing constant is finite and strictly positive, i.e., $0 < Z < \infty$.

Under these conditions, the KDE-induced tilted distribution is consistent for the true tilted distribution in total variation distance:

$$\|\hat{q}^*_\sigma - q^*\|_1 \xrightarrow{p} 0, \qquad \text{as } \sigma \to 0 \text{ and } N\sigma^d \to \infty. \tag{22}$$

Proof. By standard L^1-consistency results for kernel density estimators, we have

$$\|\hat{p}_\sigma - p\|_1 = \int |\hat{p}_\sigma(x) - p(x)|\,dx \xrightarrow{p} 0, \tag{23}$$

as $\sigma \to 0$ and $N\sigma^d \to \infty$ [18, 96].

We first show that the normalizing constant of the surrogate target distribution converges to that of the true target distribution. By (23) and the boundedness of w, we have

$$|\hat{Z}_\sigma - Z| = \left| \int (\hat{p}_\sigma(x) - p(x))\, w(x)\,dx \right| \tag{24}$$

$$\leq \|w\|_\infty \int |\hat{p}_\sigma(x) - p(x)|\,dx \tag{25}$$

$$= \|w\|_\infty \|\hat{p}_\sigma - p\|_1 \xrightarrow{p} 0. \tag{26}$$

Therefore,

$$\hat{Z}_\sigma \xrightarrow{p} Z. \tag{27}$$

We now bound the L^1 distance between the surrogate and true tilted distributions:

$$\|\hat{q}^*_\sigma - q^*\|_1 = \int \left| \frac{\hat{p}_\sigma(x)w(x)}{\hat{Z}_\sigma} - \frac{p(x)w(x)}{Z} \right| dx \tag{28}$$

$$\leq \int \left| \frac{\hat{p}_\sigma(x)w(x)}{\hat{Z}_\sigma} - \frac{p(x)w(x)}{\hat{Z}_\sigma} \right| dx + \int \left| \frac{p(x)w(x)}{\hat{Z}_\sigma} - \frac{p(x)w(x)}{Z} \right| dx \tag{29}$$

$$= \frac{1}{\hat{Z}_\sigma} \int w(x)|\hat{p}_\sigma(x) - p(x)|\,dx + \left| \frac{1}{\hat{Z}_\sigma} - \frac{1}{Z} \right| \int p(x)w(x)\,dx \tag{30}$$

$$\leq \frac{\|w\|_\infty}{\hat{Z}_\sigma} \|\hat{p}_\sigma - p\|_1 + Z \left| \frac{1}{\hat{Z}_\sigma} - \frac{1}{Z} \right|. \tag{31}$$

By (27), $\hat{Z}_\sigma \xrightarrow{p} Z > 0$. Hence, $1/\hat{Z}_\sigma = O_p(1)$. Together with (23), the first term in (31) converges to zero in probability. The second term also converges to zero by (27) and the continuous mapping theorem applied to the map $z \mapsto 1/z$, which is continuous at $Z > 0$. Therefore,

$$\|\hat{q}^*_\sigma - q^*\|_1 \xrightarrow{p} 0. \tag{32}$$

Since the total variation distance satisfies $d_{\mathrm{TV}}(\hat{q}^*_\sigma, q^*) = \frac{1}{2}\|\hat{q}^*_\sigma - q^*\|_1$, this proves that the KDE-induced surrogate target distribution $\hat{q}^*_\sigma$ consistently recovers the true reward-tilted target distribution q^* in total variation distance.

C FAV for black-box rewards

In this section, we describe how FAV can be extended to black-box rewards. Recall that the optimal Stein velocity field:

$$\phi^*_{q_\theta,q^*}(x) = \mathbb{E}_{x'\sim q_\theta}\Big[\underbrace{k(x',x)\nabla_{x'}\log p_{\text{ref}}(x')}_{\text{Prior Alignment}} + \underbrace{\beta\cdot k(x',x)\nabla_{x'}r(x')}_{\text{Reward Guidance}} + \underbrace{\nabla_{x'}k(x',x)}_{\text{Diversity Enforcement}}\Big] \tag{33}$$

requires the first-order gradient $\nabla_{x'}r(x')$ of the reward function. In the case of a black-box reward function, we replace the exact gradient with a zeroth-order estimator based on Gaussian smoothing. Specifically, given a smoothing scale $\eta > 0$, we approximate the gradient of the Gaussian-smoothed reward $\hat{r}_\eta(x') := \mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}[r(x'+\eta\epsilon)]$, which admits the closed-form expression:

$$\nabla_{x'}\hat{r}_\eta(x') = \frac{1}{\eta}\,\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}[r(x'+\eta\epsilon)\,\epsilon]. \tag{34}$$

We estimate this expectation using a two-point zeroth-order estimator with symmetric perturbations [68]:

$$\nabla_{x'}\hat{r}_\eta(x') \approx \frac{1}{K}\sum_{k=1}^{K}\frac{r(x'+\eta u_k) - r(x'-\eta u_k)}{2\eta}\,u_k, \quad u_k\sim\mathcal{N}(0,I). \tag{35}$$

This estimator only requires forward evaluations of r, making it applicable to arbitrary black-box rewards. We substitute this estimate $\nabla_{x'}\hat{r}_\eta(x')$ for $\nabla_{x'}r(x')$ and proceed with the standard FAV training objective.

In practice, the choice of perturbation space is important for reducing the variance of the zeroth-order estimate in high-dimensional data, such as images. Whenever possible, we apply the symmetric perturbations in the pre-trained representation space used by FAV. When perturbing the representation space is not feasible, we instead apply the perturbations in the latent space produced by the generative model p_{ref} before VAE decoding. This avoids performing zeroth-order estimation in the raw pixel space, where the estimator can suffer from large variance.

To validate the effectiveness of our zeroth-order gradient estimator, we conduct experiments using iMeanFlow (4-step) on ImageNet256 with three target rewards: aesthetic score, compressibility, and incompressibility. For the aesthetic score, which is differentiable but can also be treated as a black-box reward, we compare three settings: (i) FAV (Ours), which uses the exact first-order gradient; (ii) FAV-B, which uses the zeroth-order estimator in Equation 35 with $\eta = 0.005$ and $K = 16$; and (iii) Flow-GRPO+KL, a representative black-box reward optimization baseline. In this setting, perturbations and FAV training are performed in the pre-trained CLIP space used by the aesthetic score. For compressibility and incompressibility, which are fully black-box rewards, we compare only FAV-B and Flow-GRPO+KL, and perform perturbations and training in the latent space prior to VAE decoding.

As shown in Figure 6, FAV-B improves the target reward using only forward evaluations of r. Although it is less efficient than first-order FAV when reward gradients are available, it achieves substantially higher reward than Flow-GRPO+KL for all settings within the same wall time, demonstrating FAV's versatility in black-box reward settings.

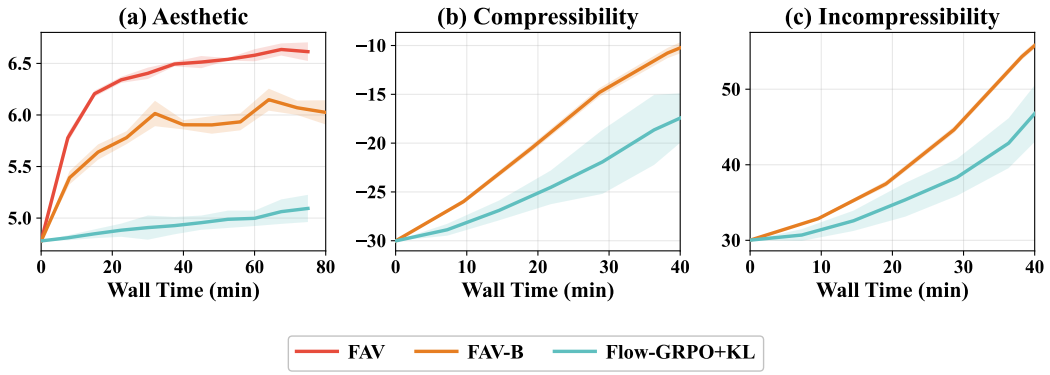

Figure 6: **FAV with non-differentiable reward.** (a) For aesthetic-score optimization, all methods are trained for 200 steps. (b),(c) For compressibility and incompressibility, FAV-B is trained for 40 steps, while Flow-GRPO+KL is run for the same wall-clock time, corresponding to 120 steps. Wall-clock time is measured on 4 NVIDIA RTX 3090 GPUs.

D Full result of RL tasks

In this section, we present the full result tables for the offline RL and offline-to-online RL experiments. Table 11 reports the complete offline RL results across 50 OGBench tasks and 6 D4RL AntMaze tasks. DSRL results on OGBench are omitted because the original paper [98] does not provide per-task performance. Table 12 reports the complete offline-to-online RL results, while Section D presents the training curves of FAV and the baselines in the offline-to-online setting.

Table 11: **Offline RL performance.**

	Gaussian Policies		Latent Opt.	Flow-based Policies			Distillation Policies			Ours	
Task	IQL	ReBRAC	DSRL	FAWAC	IFQL	QAM	SRPO	CAC	FQL	FAV-Adaptive	FAV
antmaze-large-navigate-singletask-task1-v0	48 ± 9	91 ± 10	–	1 ± 1	24 ± 17	84 ± 1	0 ± 0	42 ± 7	80 ± 8	80 ± 5	91 ± 3
antmaze-large-navigate-singletask-task2-v0	42 ± 6	88 ± 4	–	0 ± 1	8 ± 3	70 ± 11	4 ± 4	1 ± 1	57 ± 10	68 ± 6	78 ± 5
antmaze-large-navigate-singletask-task3-v0	72 ± 7	51 ± 18	–	12 ± 4	52 ± 17	89 ± 2	3 ± 2	49 ± 10	93 ± 3	90 ± 2	94 ± 1
antmaze-large-navigate-singletask-task4-v0	51 ± 9	84 ± 7	–	10 ± 3	18 ± 8	54 ± 26	45 ± 19	17 ± 6	80 ± 4	81 ± 4	85 ± 2
antmaze-large-navigate-singletask-task5-v0	54 ± 22	90 ± 2	–	9 ± 5	38 ± 18	86 ± 2	1 ± 1	55 ± 6	83 ± 4	80 ± 4	88 ± 3
antmaze-giant-navigate-singletask-task1-v0	0 ± 0	27 ± 22	–	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	4 ± 5	5 ± 4	5 ± 3
antmaze-giant-navigate-singletask-task2-v0	1 ± 1	16 ± 17	–	0 ± 0	0 ± 0	1 ± 1	0 ± 0	0 ± 0	9 ± 7	28 ± 5	28 ± 16
antmaze-giant-navigate-singletask-task3-v0	0 ± 0	34 ± 22	–	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 1	2 ± 1	3 ± 3
antmaze-giant-navigate-singletask-task4-v0	0 ± 0	5 ± 12	–	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	14 ± 23	29 ± 8	25 ± 13
antmaze-giant-navigate-singletask-task5-v0	19 ± 7	49 ± 22	–	0 ± 0	13 ± 9	13 ± 4	0 ± 0	0 ± 0	16 ± 28	68 ± 5	66 ± 12
humanoidmaze-medium-navigate-singletask-task1-v0	32 ± 7	16 ± 9	–	6 ± 2	69 ± 19	27 ± 10	0 ± 0	38 ± 19	19 ± 12	19 ± 13	30 ± 24
humanoidmaze-medium-navigate-singletask-task2-v0	41 ± 9	18 ± 16	–	40 ± 2	85 ± 11	96 ± 6	1 ± 1	47 ± 35	94 ± 3	67 ± 24	82 ± 10
humanoidmaze-medium-navigate-singletask-task3-v0	25 ± 5	36 ± 13	–	19 ± 2	49 ± 49	92 ± 3	2 ± 1	83 ± 18	74 ± 18	36 ± 30	81 ± 21
humanoidmaze-medium-navigate-singletask-task4-v0	0 ± 1	15 ± 16	–	1 ± 1	1 ± 1	1 ± 1	1 ± 1	5 ± 4	3 ± 4	12 ± 10	32 ± 23
humanoidmaze-medium-navigate-singletask-task5-v0	66 ± 4	24 ± 20	–	31 ± 7	98 ± 2	99 ± 1	3 ± 3	91 ± 5	97 ± 2	84 ± 31	96 ± 2
humanoidmaze-large-navigate-singletask-task1-v0	3 ± 1	2 ± 1	–	0 ± 0	6 ± 2	3 ± 2	0 ± 0	1 ± 1	7 ± 6	5 ± 3	10 ± 10
humanoidmaze-large-navigate-singletask-task2-v0	0 ± 0	0 ± 0	–	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0
humanoidmaze-large-navigate-singletask-task3-v0	7 ± 3	8 ± 4	–	1 ± 1	48 ± 10	9 ± 4	1 ± 1	2 ± 3	11 ± 7	12 ± 6	14 ± 10
humanoidmaze-large-navigate-singletask-task4-v0	1 ± 0	1 ± 1	–	0 ± 0	1 ± 1	2 ± 3	0 ± 0	0 ± 1	2 ± 3	1 ± 1	2 ± 3
humanoidmaze-large-navigate-singletask-task5-v0	1 ± 1	2 ± 2	–	0 ± 0	0 ± 0	5 ± 11	0 ± 0	0 ± 0	1 ± 3	0 ± 0	1 ± 1
antsoccer-arena-navigate-singletask-task1-v0	14 ± 5	0 ± 0	–	22 ± 2	61 ± 25	92 ± 1	2 ± 1	1 ± 3	77 ± 4	79 ± 5	82 ± 3
antsoccer-arena-navigate-singletask-task2-v0	17 ± 7	0 ± 1	–	8 ± 1	75 ± 3	97 ± 2	3 ± 1	0 ± 0	88 ± 3	86 ± 3	92 ± 2
antsoccer-arena-navigate-singletask-task3-v0	6 ± 4	0 ± 0	–	11 ± 5	14 ± 22	52 ± 6	0 ± 0	8 ± 19	61 ± 6	57 ± 6	53 ± 5
antsoccer-arena-navigate-singletask-task4-v0	3 ± 2	0 ± 0	–	12 ± 3	16 ± 9	40 ± 9	0 ± 0	0 ± 0	39 ± 6	41 ± 6	41 ± 6
antsoccer-arena-navigate-singletask-task5-v0	2 ± 2	0 ± 0	–	9 ± 2	0 ± 1	25 ± 6	0 ± 0	0 ± 0	36 ± 9	40 ± 16	28 ± 6
cube-single-play-singletask-task1-v0	88 ± 3	89 ± 5	–	81 ± 9	79 ± 4	53 ± 30	89 ± 7	77 ± 28	97 ± 2	94 ± 2	94 ± 2
cube-single-play-singletask-task2-v0	85 ± 8	92 ± 4	–	81 ± 9	73 ± 3	57 ± 10	82 ± 16	80 ± 30	97 ± 2	93 ± 2	95 ± 2
cube-single-play-singletask-task3-v0	91 ± 5	93 ± 3	–	87 ± 4	88 ± 4	64 ± 30	96 ± 2	98 ± 1	98 ± 2	96 ± 2	98 ± 1
cube-single-play-singletask-task4-v0	73 ± 6	92 ± 3	–	79 ± 6	79 ± 6	61 ± 10	70 ± 18	91 ± 2	94 ± 3	91 ± 4	91 ± 2
cube-single-play-singletask-task5-v0	78 ± 9	87 ± 8	–	78 ± 10	77 ± 7	50 ± 19	61 ± 12	80 ± 20	93 ± 3	85 ± 4	86 ± 2
cube-double-play-singletask-task1-v0	27 ± 5	45 ± 6	–	21 ± 7	35 ± 9	38 ± 20	7 ± 6	21 ± 8	61 ± 9	43 ± 6	47 ± 5
cube-double-play-singletask-task2-v0	1 ± 1	7 ± 3	–	2 ± 1	9 ± 6	27 ± 12	0 ± 0	2 ± 2	36 ± 6	23 ± 6	21 ± 3
cube-double-play-singletask-task3-v0	0 ± 0	4 ± 1	–	1 ± 1	8 ± 6	28 ± 18	0 ± 1	3 ± 1	22 ± 6	12 ± 3	12 ± 5
cube-double-play-singletask-task4-v0	0 ± 0	1 ± 1	–	0 ± 0	1 ± 1	12 ± 6	0 ± 0	0 ± 1	5 ± 2	5 ± 2	4 ± 1
cube-double-play-singletask-task5-v0	4 ± 3	4 ± 2	–	2 ± 1	17 ± 6	46 ± 16	0 ± 0	3 ± 2	19 ± 10	38 ± 13	44 ± 9
scene-play-singletask-task1-v0	94 ± 3	95 ± 2	–	87 ± 8	98 ± 3	100 ± 0	94 ± 4	100 ± 1	100 ± 0	97 ± 2	98 ± 2
scene-play-singletask-task2-v0	12 ± 3	50 ± 13	–	18 ± 8	0 ± 0	96 ± 2	2 ± 2	50 ± 40	76 ± 9	87 ± 7	88 ± 4
scene-play-singletask-task3-v0	32 ± 7	55 ± 16	–	38 ± 9	54 ± 19	97 ± 1	4 ± 4	49 ± 16	98 ± 1	91 ± 4	88 ± 4
scene-play-singletask-task4-v0	0 ± 1	3 ± 3	–	6 ± 1	0 ± 0	0 ± 1	0 ± 0	0 ± 0	5 ± 1	1 ± 2	0 ± 1
scene-play-singletask-task5-v0	0 ± 0	0 ± 0	–	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0
puzzle-3x3-play-singletask-task1-v0	33 ± 6	97 ± 4	–	25 ± 9	94 ± 3	84 ± 32	89 ± 5	97 ± 2	90 ± 4	97 ± 3	99 ± 2
puzzle-3x3-play-singletask-task2-v0	4 ± 3	1 ± 1	–	4 ± 2	1 ± 2	4 ± 4	0 ± 1	0 ± 0	16 ± 5	56 ± 26	74 ± 18
puzzle-3x3-play-singletask-task3-v0	3 ± 2	3 ± 1	–	1 ± 0	0 ± 0	5 ± 6	0 ± 0	0 ± 0	10 ± 3	25 ± 10	61 ± 13
puzzle-3x3-play-singletask-task4-v0	2 ± 1	2 ± 1	–	1 ± 1	0 ± 0	2 ± 2	0 ± 0	0 ± 0	16 ± 5	65 ± 20	56 ± 15
puzzle-3x3-play-singletask-task5-v0	3 ± 2	5 ± 3	–	1 ± 1	0 ± 0	2 ± 1	0 ± 0	0 ± 0	16 ± 3	74 ± 15	76 ± 14
puzzle-4x4-play-singletask-task1-v0	12 ± 2	26 ± 4	–	1 ± 2	49 ± 9	73 ± 7	24 ± 9	44 ± 10	34 ± 8	28 ± 5	45 ± 11
puzzle-4x4-play-singletask-task2-v0	7 ± 4	12 ± 4	–	0 ± 1	4 ± 4	21 ± 3	0 ± 1	0 ± 0	16 ± 5	5 ± 3	3 ± 3
puzzle-4x4-play-singletask-task3-v0	9 ± 3	15 ± 3	–	1 ± 1	50 ± 14	62 ± 5	21 ± 10	29 ± 12	18 ± 5	15 ± 3	16 ± 10
puzzle-4x4-play-singletask-task4-v0	5 ± 2	10 ± 3	–	0 ± 0	21 ± 11	21 ± 2	7 ± 4	1 ± 1	11 ± 3	7 ± 2	10 ± 6
puzzle-4x4-play-singletask-task5-v0	4 ± 1	7 ± 3	–	0 ± 1	2 ± 2	12 ± 3	1 ± 1	0 ± 0	7 ± 3	3 ± 2	4 ± 5
antmaze-umaze-v2	77	98	94 ± 2	90 ± 6	92 ± 6	95 ± 1	97	66 ± 5	96 ± 2	96 ± 2	94 ± 2
antmaze-umaze-diverse-v2	54	84	61 ± 5	55 ± 7	62 ± 12	86 ± 5	82	66 ± 11	89 ± 5	89 ± 3	86 ± 5
antmaze-medium-play-v2	66	90	66 ± 5	52 ± 12	56 ± 15	83 ± 3	81	49 ± 24	78 ± 7	74 ± 6	81 ± 6
antmaze-medium-diverse-v2	74	84	20 ± 6	44 ± 15	60 ± 25	78 ± 5	75	0 ± 1	71 ± 13	64 ± 8	67 ± 6
antmaze-large-play-v2	42	52	43 ± 6	10 ± 6	55 ± 9	63 ± 24	54	0 ± 0	84 ± 7	72 ± 6	75 ± 10
antmaze-large-diverse-v2	30	64	54 ± 8	16 ± 10	64 ± 8	72 ± 6	54	0 ± 0	83 ± 4	76 ± 1	78 ± 2

Table 12: **Offline-to-Online RL performance.** The left value denotes performance after 1M offline training steps, and the right value denotes performance after an additional 1M steps of online fine-tuning.

Task (offline 1M → online 1M)	IQL	ReBRAC	RLPD	IFQL	QAM	FQL	FAV
antmaze-giant-navigate-singletask-task1-v0	0 ± 0 → 0 ± 0	52 ± 16 → 99 ± 1	0 ± 0 → 14 ± 23	0 ± 0 → 0 ± 0	0 ± 0 → 0 ± 0	0 ± 1 → 30 ± 40	4 ± 6 → 68 ± 8
antmaze-giant-navigate-singletask-task2-v0	0 ± 0 → 1 ± 2	15 ± 11 → 99 ± 1	0 ± 0 → 67 ± 26	0 ± 0 → 0 ± 0	1 ± 1 → 5 ± 3	0 ± 0 → 96 ± 2	29 ± 18 → 88 ± 5
antmaze-giant-navigate-singletask-task3-v0	0 ± 0 → 0 ± 0	32 ± 32 → 98 ± 3	0 ± 0 → 81 ± 14	0 ± 0 → 0 ± 0	0 ± 0 → 0 ± 0	0 ± 0 → 6 ± 15	5 ± 5 → 34 ± 14
antmaze-giant-navigate-singletask-task4-v0	0 ± 0 → 0 ± 1	2 ± 5 → 100 ± 0	0 ± 0 → 77 ± 30	0 ± 0 → 0 ± 0	0 ± 0 → 1 ± 1	0 ± 0 → 2 ± 4	30 ± 13 → 86 ± 3
antmaze-giant-navigate-singletask-task5-v0	20 ± 7 → 13 ± 7	52 ± 15 → 99 ± 1	0 ± 0 → 82 ± 31	10 ± 9 → 0 ± 0	18 ± 4 → 48 ± 29	0 ± 0 → 85 ± 32	72 ± 10 → 96 ± 2
antsoccer-arena-navigate-singletask-task1-v0	17 ± 7 → 10 ± 4	0 ± 0 → 0 ± 0	0 ± 0 → 74 ± 8	65 ± 20 → 85 ± 5	93 ± 3 → 97 ± 3	78 ± 7 → 97 ± 2	84 ± 7 → 99 ± 1
antsoccer-arena-navigate-singletask-task2-v0	14 ± 7 → 7 ± 4	0 ± 1 → 0 ± 0	0 ± 0 → 75 ± 9	54 ± 32 → 73 ± 31	97 ± 2 → 99 ± 1	89 ± 5 → 95 ± 3	93 ± 3 → 97 ± 1
antsoccer-arena-navigate-singletask-task3-v0	5 ± 7 → 2 ± 2	0 ± 0 → 0 ± 0	0 ± 0 → 28 ± 14	8 ± 14 → 48 ± 21	54 ± 9 → 96 ± 3	51 ± 11 → 91 ± 7	54 ± 9 → 92 ± 4
antsoccer-arena-navigate-singletask-task4-v0	4 ± 4 → 0 ± 1	0 ± 0 → 0 ± 0	0 ± 0 → 0 ± 0	8 ± 10 → 30 ± 18	38 ± 15 → 84 ± 20	37 ± 11 → 81 ± 13	44 ± 14 → 82 ± 6
antsoccer-arena-navigate-singletask-task5-v0	3 ± 1 → 1 ± 1	0 ± 0 → 0 ± 0	0 ± 0 → 6 ± 9	0 ± 1 → 7 ± 10	24 ± 8 → 88 ± 6	27 ± 11 → 80 ± 17	34 ± 11 → 89 ± 4
cube-double-play-singletask-task1-v0	10 ± 3 → 1 ± 1	21 ± 8 → 100 ± 0	0 ± 0 → 12 ± 11	21 ± 4 → 92 ± 4	31 ± 26 → 24 ± 42	58 ± 6 → 99 ± 3	50 ± 10 → 99 ± 1
cube-double-play-singletask-task2-v0	0 ± 1 → 0 ± 0	4 ± 3 → 24 ± 18	0 ± 0 → 0 ± 0	8 ± 3 → 58 ± 15	30 ± 13 → 12 ± 32	32 ± 9 → 92 ± 5	20 ± 8 → 98 ± 1
cube-double-play-singletask-task3-v0	0 ± 0 → 0 ± 0	4 ± 2 → 44 ± 26	0 ± 0 → 0 ± 0	4 ± 2 → 44 ± 17	26 ± 20 → 37 ± 48	21 ± 9 → 90 ± 12	11 ± 6 → 98 ± 2
cube-double-play-singletask-task4-v0	0 ± 0 → 0 ± 0	1 ± 2 → 0 ± 1	0 ± 0 → 0 ± 0	0 ± 0 → 0 ± 1	10 ± 6 → 22 ± 30	7 ± 3 → 5 ± 6	3 ± 2 → 64 ± 11
cube-double-play-singletask-task5-v0	1 ± 1 → 0 ± 0	2 ± 2 → 8 ± 6	0 ± 0 → 0 ± 0	12 ± 6 → 77 ± 13	32 ± 28 → 24 ± 42	16 ± 9 → 92 ± 4	41 ± 11 → 96 ± 4
humanoidmaze-large-navigate-singletask-task1-v0	1 ± 1 → 0 ± 1	1 ± 1 → 0 ± 0	0 ± 0 → 0 ± 0	8 ± 4 → 1 ± 2	2 ± 3 → 0 ± 0	5 ± 7 → 0 ± 0	11 ± 12 → 18 ± 19
humanoidmaze-large-navigate-singletask-task2-v0	0 ± 0 → 0 ± 0	0 ± 1 → 0 ± 0	0 ± 0 → 0 ± 0	0 ± 0 → 0 ± 0	0 ± 0 → 0 ± 0	0 ± 0 → 0 ± 0	0 ± 1 → 1 ± 2
humanoidmaze-large-navigate-singletask-task3-v0	5 ± 4 → 7 ± 3	4 ± 6 → 0 ± 0	0 ± 0 → 0 ± 0	52 ± 11 → 36 ± 17	12 ± 7 → 50 ± 31	17 ± 6 → 16 ± 21	14 ± 10 → 20 ± 27
humanoidmaze-large-navigate-singletask-task4-v0	0 ± 1 → 0 ± 1	1 ± 1 → 1 ± 1	0 ± 0 → 0 ± 0	0 ± 0 → 12 ± 10	2 ± 3 → 32 ± 22	1 ± 1 → 5 ± 10	2 ± 3 → 0 ± 1
humanoidmaze-large-navigate-singletask-task5-v0	3 ± 3 → 1 ± 1	0 ± 1 → 0 ± 0	0 ± 0 → 0 ± 0	2 ± 4 → 9 ± 9	5 ± 11 → 23 ± 28	0 ± 1 → 0 ± 0	1 ± 2 → 0 ± 0
puzzle-4x4-play-singletask-task1-v0	3 ± 3 → 0 ± 1	26 ± 6 → 100 ± 0	0 ± 0 → 100 ± 1	37 ± 4 → 99 ± 2	75 ± 8 → 62 ± 48	23 ± 10 → 100 ± 0	55 ± 14 → 100 ± 0
puzzle-4x4-play-singletask-task2-v0	1 ± 2 → 1 ± 1	8 ± 5 → 0 ± 0	0 ± 0 → 100 ± 0	16 ± 8 → 0 ± 0	22 ± 8 → 25 ± 43	12 ± 6 → 1 ± 2	6 ± 8 → 100 ± 0
puzzle-4x4-play-singletask-task3-v0	1 ± 1 → 0 ± 1	17 ± 4 → 86 ± 31	0 ± 0 → 100 ± 0	51 ± 13 → 99 ± 1	60 ± 4 → 62 ± 48	14 ± 7 → 99 ± 3	18 ± 8 → 100 ± 0
puzzle-4x4-play-singletask-task4-v0	2 ± 2 → 0 ± 0	7 ± 3 → 2 ± 3	0 ± 0 → 100 ± 0	21 ± 4 → 27 ± 40	19 ± 6 → 50 ± 50	7 ± 5 → 50 ± 50	8 ± 5 → 100 ± 0
puzzle-4x4-play-singletask-task5-v0	0 ± 1 → 1 ± 2	8 ± 3 → 0 ± 0	0 ± 0 → 100 ± 0	18 ± 6 → 0 ± 0	12 ± 7 → 0 ± 0	6 ± 3 → 0 ± 0	3 ± 5 → 25 ± 43
scene-play-singletask-task1-v0	66 ± 7 → 99 ± 1	94 ± 4 → 100 ± 0	0 ± 0 → 100 ± 0	100 ± 1 → 100 ± 0	100 ± 0 → 100 ± 0	100 ± 0 → 100 ± 0	98 ± 3 → 100 ± 0
scene-play-singletask-task2-v0	18 ± 11 → 53 ± 21	58 ± 14 → 100 ± 0	0 ± 0 → 100 ± 1	53 ± 22 → 95 ± 3	96 ± 4 → 100 ± 0	77 ± 11 → 100 ± 0	86 ± 4 → 100 ± 0
scene-play-singletask-task3-v0	17 ± 7 → 14 ± 7	53 ± 15 → 100 ± 0	0 ± 0 → 96 ± 1	81 ± 4 → 90 ± 6	98 ± 4 → 99 ± 2	94 ± 5 → 100 ± 1	88 ± 8 → 100 ± 0
scene-play-singletask-task4-v0	1 ± 1 → 0 ± 0	0 ± 1 → 0 ± 0	0 ± 0 → 0 ± 0	0 ± 1 → 0 ± 0	0 ± 1 → 0 ± 0	5 ± 5 → 0 ± 0	0 ± 1 → 8 ± 21
scene-play-singletask-task5-v0	0 ± 1 → 0 ± 0	0 ± 0 → 0 ± 0	0 ± 0 → 0 ± 0	0 ± 0 → 0 ± 0	0 ± 0 → 0 ± 0	0 ± 0 → 0 ± 0	0 ± 0 → 0 ± 0
OGBench Average (30 tasks)	6 ± 1 → 7 ± 1	16 ± 1 → 39 ± 2	0 ± 0 → 44 ± 2	21 ± 2 → 36 ± 2	32 ± 2 → 41 ± 6	26 ± 1 → 54 ± 2	32 ± 2 → 69 ± 2

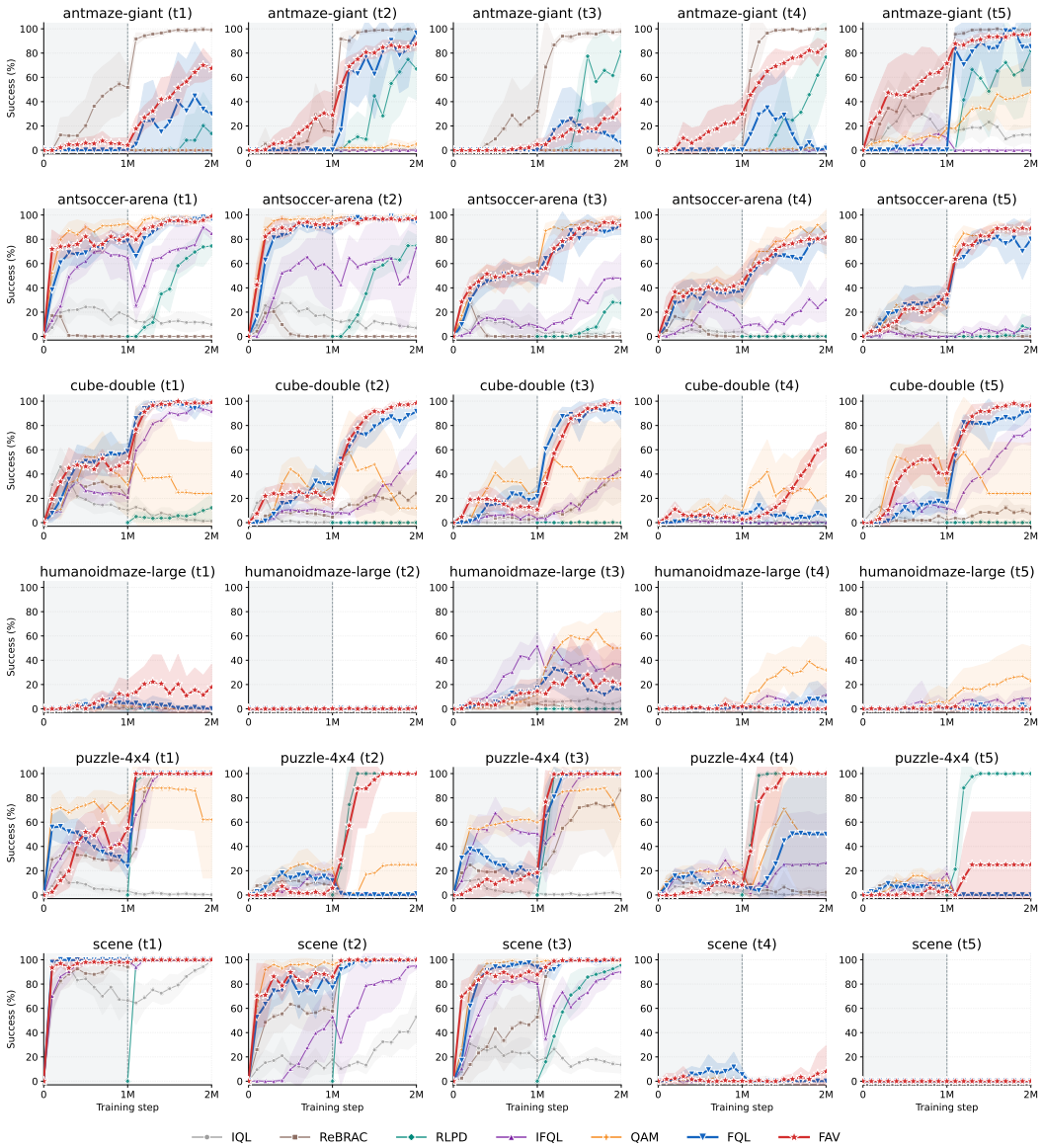

Figure 7: **Training curves for offline-to-online RL.**

E Computation analysis

We analyze the computational cost of FAV by comparing its training and inference times with baselines on RL and image-generator alignment experiments.

E.1 Offline RL

We evaluate a total of eight offline RL methods, including FAV, on the `scene-play-single` environment. To ensure a fair comparison, all methods are executed under identical configurations on a single NVIDIA RTX 3090 GPU, with results averaged over 8 random seeds.

Table 13: Training and inference time for each method on `scene-play-single`.

Method	**Training Time (ms/step)**	**Inference Time (ms/step)**
FQL	2.65	0.56
IFQL	1.96	0.85
IQL	2.08	0.60
ReBRAC	1.51	0.59
RLPD	1.14	0.51
DSRL	1.96	0.64
QAM	4.67	0.78
FAV (Ours)	3.05	0.53

E.2 Image generator alignment

We conduct a computational analysis on seven image-generator alignment algorithms, including FAV. For a fair comparison, all fine-tuning methods are evaluated with an effective batch size of 256 per step. Experiments using the iMeanFlow backbone are run on 4 NVIDIA RTX 3090 GPUs, while experiments using the SANA-Sprint backbone are run on 4 NVIDIA RTX 4090 GPUs.

Table 14: Training and inference time for each method.

Backbone	**Method**	**Reward**	**Training Time (s/step)**	**Inference Time (s/step)**
iMF (4 steps)	Flow-GRPO+KL	Aesthetic	16.43	0.1
iMF (4 steps)	Flow-GRPO+KL	Compressibility	21.53	0.1
iMF (4 steps)	DRaFT	Aesthetic	28.35	0.1
iMF (4 steps)	Adjoint Matching	Aesthetic	29.95	0.1
iMF (4 steps)	Best-of-256	Aesthetic	–	28.6
iMF (4 steps)	ReNO-50	Aesthetic	–	17.9
SANA-Sprint 1.6B (4 steps)	RLCM	HPS	23.38	0.6
SANA-Sprint 1.6B (4 steps)	RLCM	NSFW	23.14	0.6
SANA-Sprint 1.6B (4 steps)	DRaFT	HPS	40.34	0.6
SANA-Sprint 1.6B (4 steps)	DRaFT	NSFW	39.93	0.6
iMF (4 steps)	FAV (Ours)	Aesthetic	20.12	0.1
iMF (4 steps)	FAV (Ours)	Compressibility	60.14	0.1
SANA-Sprint 1.6B (4 steps)	FAV (Ours)	HPS	28.63	0.6
SANA-Sprint 1.6B (4 steps)	FAV (Ours)	NSFW	28.34	0.6

F Ablation analysis

In this section, we analyze the effect of each component in the optimal transport vector of FAV: the prior alignment term, the reward guidance term, and the repulsive term in Equation (5). As shown in Figure 8, removing the prior alignment term ("w/o prior") leads to substantially higher reward scores but causes severe mode collapse. Without the reward guidance term ("w/o reward"), FAV fails to optimize the target reward, with the aesthetic score remaining flat throughout training. Removing the repulsive term ("w/o repulsive") yields a small but consistent decrease in diversity compared to the full model, suggesting that it offers a marginal contribution to diversity preservation on top of the prior alignment term. Experiments are conducted using iMeanFlow (4 steps) on ImageNet 256 with the aesthetic score as the target reward, averaged over 4 seeds.

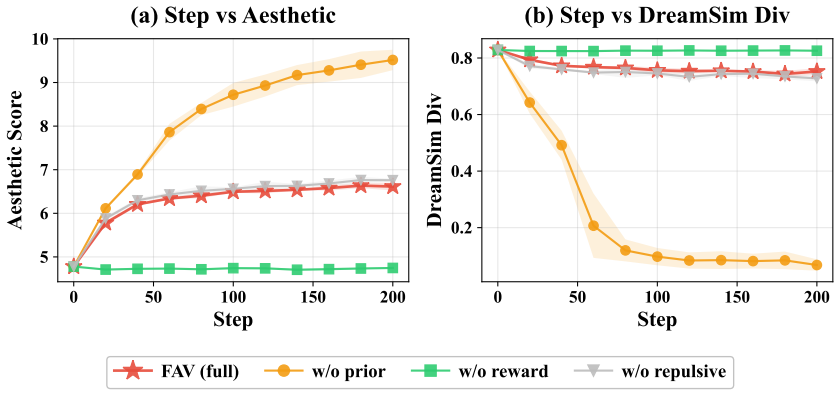

Figure 8: **Ablation analysis for each components of FAV.**

G Sensitivity analysis

In this section, we conduct a sensitivity analysis on the key hyperparameters of FAV: the temperature parameter β and the kernel bandwidth τ. Experiments are conducted using iMeanFlow (4 steps) on ImageNet 256 with the aesthetic score as the target reward, averaged over 4 seeds. As shown in Figure 9, β controls the reward alignment strength, enabling a tunable reward–diversity tradeoff: larger β pushes samples more aggressively toward high-reward modes, while smaller β stays closer to the prior.

The kernel bandwidth τ, defined as $\tau := 2\sigma^2$, controls the interaction scale of the kernel k_σ. As shown in Figure 10, increasing τ generally improves reward optimization but reduces sample diversity. We attribute this trade-off to the locality of kernel interactions. When τ is small, samples mainly interact with nearby reference points, preserving the multi-modal structure of the prior and maintaining diversity. However, such local updates limit reward improvement because samples are unlikely to move toward higher-reward modes outside their local neighborhoods. In contrast, when τ is large, the update becomes overly global, smoothing interactions across distinct modes. This weakens mode separation and makes samples more likely to collapse toward the single highest-reward mode, thereby reducing diversity.

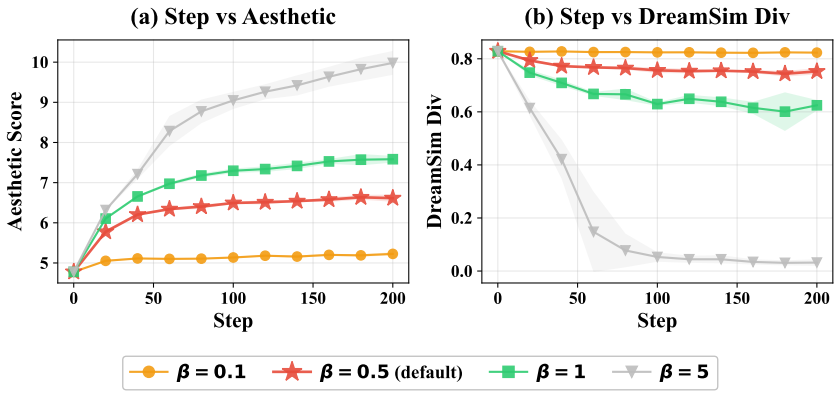

Figure 9: **Sensitivity analysis for temperature parameter β.**

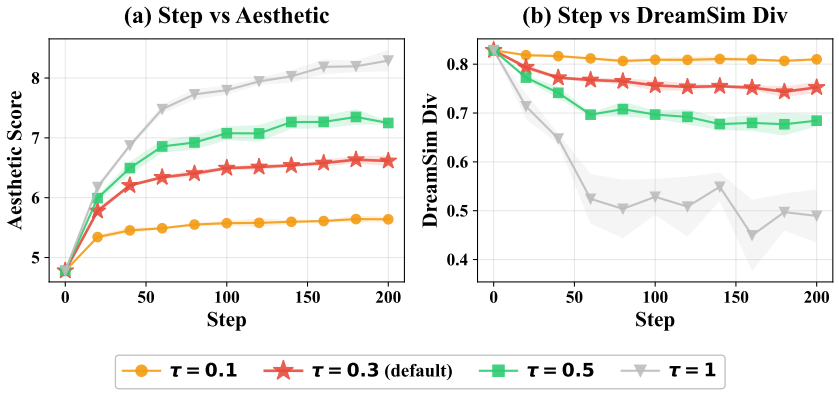

Figure 10: **Sensitivity analysis for kernel bandwidth τ.**

We additionally evaluate FAV's sensitivity on the RL side using three environments: `puzzle-3x3-play` (5 tasks; center $\tau^\star$=0.5, $\beta^\star$=0.5), D4RL `antmaze-large-play` (1 task; $\tau^\star$=1, $\beta^\star$=3), and `antsoccer-arena-navigate` (5 tasks; $\tau^\star$=1, $\beta^\star$=1). For each environment, we perform two axis-aligned 1D sweeps about its center: one varies $\beta \in \{0.5, 1, 2, 3\}$ with τ=$\tau^\star$, the other varies $\tau \in \{0.05, 0.1, 0.5, 1\}$ with β=$\beta^\star$. Each cell of Table 15 reports mean $\pm$ std over 4 seeds at 1M offline gradient steps; centers are marked * and per-column best in bold.

The sensitivity study shows that both β and τ are important hyperparameters for FAV. In particular, too small τ can substantially degrade performance, as very small values make the kernel nearly one-hot and lead to collapse. However, this concern is mitigated in practice by FAV-Adaptive, where τ is selected automatically using Scott's rule, as demonstrated in our main experiments. This leaves β, which controls the strength of the Q-gradient in the reward-tilted transport, as the primary hyperparameter to tune. We note that such a reward-guidance

Table 15: **RL hyperparameter sensitivity.**

β	puzzle-3x3	antmaze-l	antsoccer	τ	puzzle-3x3	antmaze-l	antsoccer
0.5	$\mathbf{73.4 \pm 23.4}^\star$	2.0 ± 2.8	54.9 ± 13.1	0.05	22.9 ± 39.5	0.0 ± 0.0	0.0 ± 0.0
1.0	64.3 ± 29.4	30.0 ± 11.2	$\mathbf{60.0 \pm 24.3}^\star$	0.10	21.3 ± 39.5	0.0 ± 0.0	0.0 ± 0.0
2.0	27.6 ± 31.0	$\mathbf{64.5 \pm 6.6}$	39.6 ± 31.9	0.50	$\mathbf{73.4 \pm 23.4}^\star$	$\mathbf{62.5 \pm 3.4}$	54.4 ± 12.2
3.0	14.7 ± 22.5	$58.0 \pm 38.7^\star$	17.1 ± 18.0	1.00	61.0 ± 24.1	$58.0 \pm 38.7^\star$	$\mathbf{60.0 \pm 24.3}^\star$

coefficient is not unique to FAV: analogous hyperparameters appear in most baselines [20, 73, 50]. Despite using a relatively small search range of β, FAV achieves strong performance across environments, suggesting that its sensitivity remains manageable in practice.

H High-resolution text-to-image alignment results

In this section, we compare fine-tuning methods (DRaFT [14], RLCM [70]) with FAV on high-resolution text-to-image generation. Experiments are conducted using Sana-Sprint 1.6B (4 steps) with the NSFW classifier and HPS as target rewards. As shown in Figure 11, FAV achieves the highest target reward while preserving the image quality and diversity.

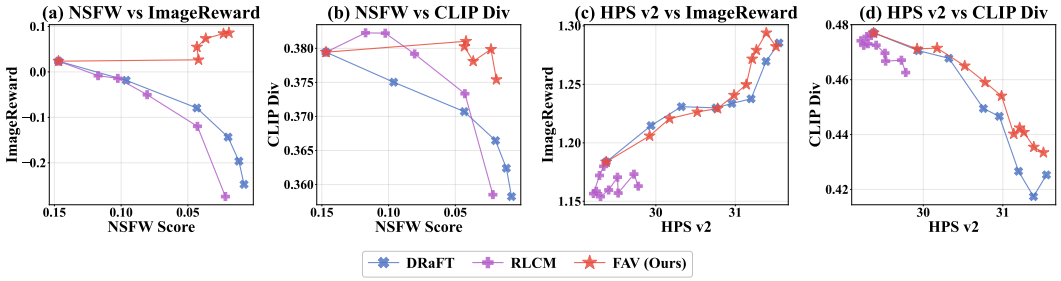

Figure 11: **Training dynamics of each alignment method.** NSFW classifier as the target reward for (a),(b); HPS as the target reward for (c),(d).

I Qualitative analysis

In this section, we provide qualitative results of the image generator alignment experiments.

I.1 ImageNet-256

Figure 12 shows qualitative comparisons on ImageNet 256, where each method aligns the iMeanFlow (8 steps) model with the aesthetic score as the target reward.



Figure 12: **Qualitative comparison on ImageNet 256**.

I.2 High-resolution text-to-image generation

Figure 13 shows qualitative samples from Sana-Sprint 1.6B (4-step) fine-tuned with each alignment method. The first four rows show samples from models fine-tuned on DrawBench prompts to maximize the HPS reward, and the last row shows samples from models fine-tuned on Sneaky prompts to minimize the NSFW classifier score. Since Sneaky prompts contain adversarial prompt and cannot be disclosed here, we refer readers to the official repository[3] for the prompt used in the last row.

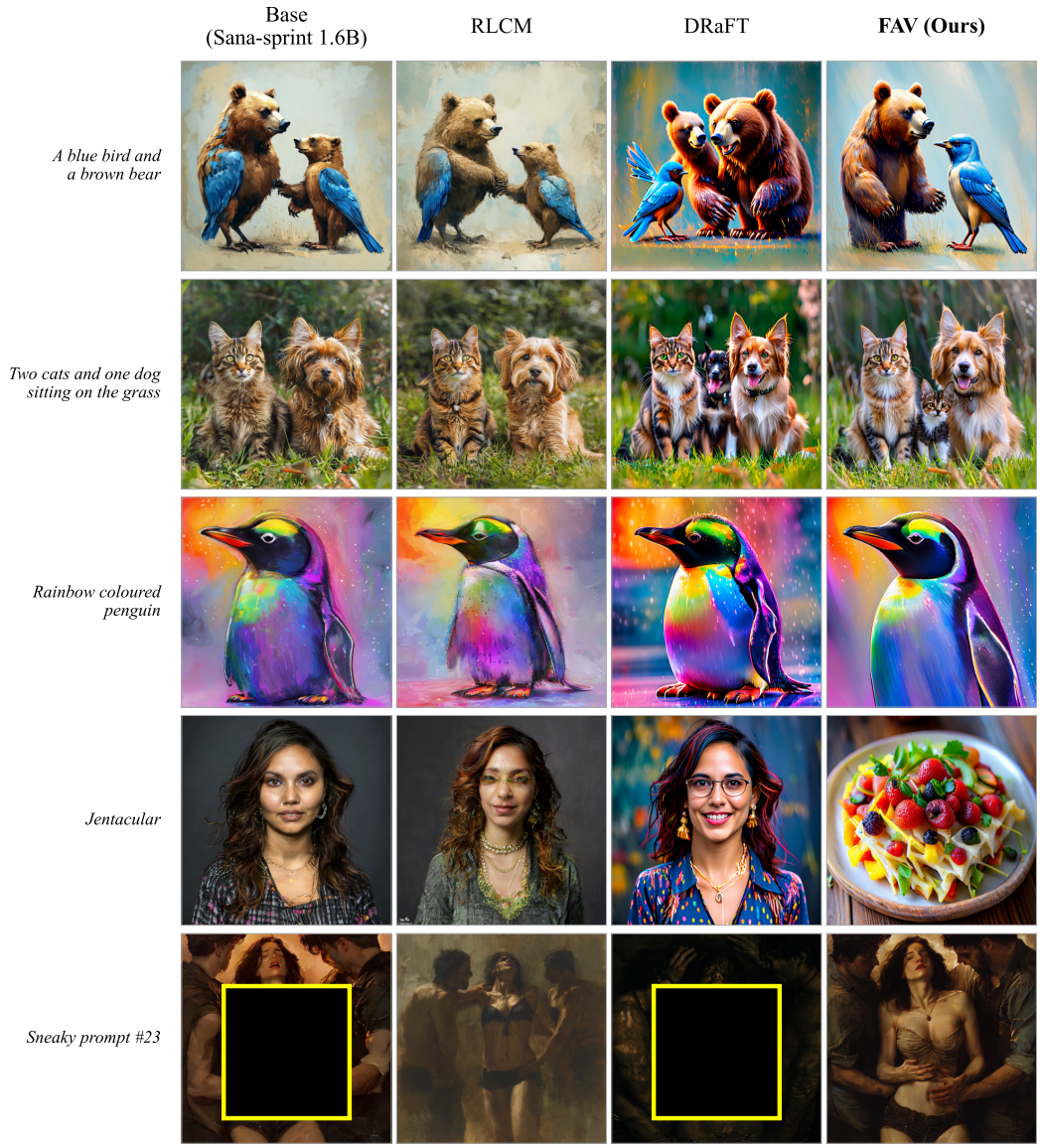

Figure 13: **Qualitative results for high-resolution text-to-image generation.**

[3] https://github.com/Yuchen413/text2image_safety

J Broader impacts

Sampling from reward-tilted distributions enables flexible alignment of generative models to diverse downstream objectives, but it also carries the risk of unintended or harmful outcomes. For example, while reward tilting can be used to steer models toward safer and more helpful generations, the same mechanism could be misused by inverting safety-related rewards to elicit unsafe or malicious outputs. As reward-based alignment methods become increasingly powerful and accessible, it is essential for researchers to apply them responsibly and to consider the broader societal implications of their work.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state that FAV is a sample-based alignment framework requiring only samples from the generator and reference distribution, and that we evaluate it on offline/offline-to-online RL and on image generator alignment across diverse few-step backbones; these claims are directly supported by Sections 4–5 and the appendix.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The Limitations paragraph in the Discussion section explicitly states that FAV's KDE-based nonparametric score estimation is only asymptotically consistent, so finite reference samples introduce approximation error that grows in high-dimensional domains, and notes that performing transport in a pretrained representation space for image experiments only partially mitigates this.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The only theoretical result is the asymptotic consistency of the KDE-based score estimator. The required regularity conditions on p and the convergence regime ($\sigma \to 0$, $N\sigma^{d+2} \to \infty$) are stated, and a proof decomposing estimation error and smoothing bias is given in Appendix.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The FAV training objective is given in pseudocode (Algorithm 1) and equations (Equations (6) and (7)). Datasets, model backbones, reward functions, and full hyperparameter sweeps for the toy, RL (Appendix A.2), and image alignment (Appendix A.3) experiments are documented in the appendix, along with the same details for every baseline.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

Justification: Code is not released with this anonymous submission; we plan to release the implementation upon acceptance. All datasets used (OGBench, D4RL, ImageNet) and the pre-trained backbones (StyleGAN-XL, IMM, iMeanFlow, Drifting Model, Sana-Sprint) are publicly available, and the paper documents the upstream codebases (FQL, QAM) on which our RL implementation is built.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Optimizer, learning rate, batch size, architecture, evaluation protocol, the number of seeds, and the searched ranges for each FAV hyperparameter (β, τ) are reported in Appendices A.1 to A.3, with per-task selected values listed in the corresponding hyperparameter tables.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: RL results are reported as the mean over 8 random seeds with 95% confidence intervals (Tables 1 and 2). Image generator alignment results are computed over 4 seeds with 1,024 images per condition (32 prompts/labels $\times$ 32 samples) for every metric. Error-bar definitions are stated in the experimental-setup paragraphs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix E reports per-step training and inference times (ms/step) for FAV and seven baselines on a single NVIDIA RTX 3090 GPU. Wall-clock comparisons for image alignment on $4\times$ RTX 3090 are reported in Figure 6.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research uses publicly available datasets and pre-trained models in accordance with their intended use, does not involve human subjects, and includes a safety-alignment experiment that down-weights an NSFW classifier on adversarial prompts.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: FAV is a general-purpose alignment framework for steering generative models toward reward-tilted distributions. On the positive side, such reward-based alignment can improve downstream utility and safety; for example, our text-to-image experiment uses an NSFW classifier as a safety-alignment objective. On the negative side, the same mechanism could be misused by optimizing harmful or inverted safety-related rewards, potentially eliciting unsafe or malicious outputs. Thus, while FAV is not itself a deployed system or dataset, its societal impacts depend on the underlying generators, reward models, and deployment context and should be carefully considered.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release new pre-trained models, scraped datasets, or other high-risk assets; it studies an alignment algorithm applied to publicly available checkpoints.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets (OGBench, D4RL, ImageNet, DrawBench, SneakyPrompt), pre-trained generators (StyleGAN-XL, IMM, iMeanFlow, Drifting Model, Sana-Sprint), reward models (LAION aesthetic, HPSv2, ImageReward, DreamSim, CLIP, Falconsai NSFW), and the FQL/QAM upstream codebases are cited and used in accordance with their original publications and public releases (Section A.3 and the experimental setup of Appendix A.3).

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

Justification: The paper does not release new datasets, pre-trained models, or benchmarks.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects. Human-preference signals come from the pre-existing HPSv2 reward model, not from new human-subjects data collected by the authors.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve human subjects, so IRB review is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not part of the core methodology of this paper.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.