TAMING GENERATIVE CONTROL POLICIES FOR REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative control policies have emerged as an effective policy parameterization for robot learning. They leverage generative models such as diffusion and flow matching to iteratively transform noise into actions, enabling these policies to effectively model complex action distributions. However, prior work has reported that optimizing these policies with reinforcement learning (RL) can lead to training instability. This instability has often been associated with the iterative nature of the generation process. Many existing methods therefore avoid directly backpropagating value gradients through the entire iterative generation process, leading suboptimal performance. Contrary to prior belief, We show that iterative generation can support stable value-gradient RL and identify sampler design as an important factor in optimization. Motivated by this insight, we propose VINE, a new sampling strategy that reconstructs the generation path at each iteration, enabling full end-to-end value-gradient propagation. VINE requires only a few lines of changes to the vanilla flow matching sampler. VINE retains iterative generation and end-to-end value-gradient optimization through the entire sampling chain. Experiments show that VINE outperforms the evaluated full-BPTT baselines in aggregate on OGBench tasks. VINE enables stable off-policy RL adaptation of the pretrained $\pi_{0.5}$ VLA across LIBERO suites. On real-world manipulation tasks, VINE rapidly improves success rate toward 100% from scratch within 20 minutes of online rollouts. Videos are available at the website.

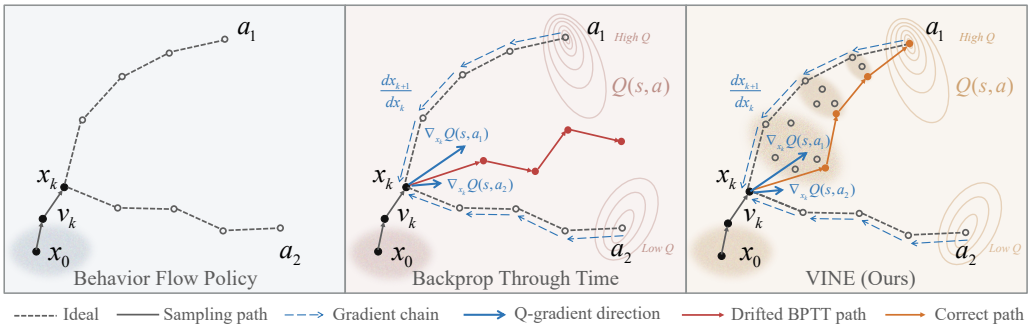

Figure 1: Flow policies stably learn multimodal action distributions (a_1 and a_2) through behavior cloning (left). However, applying value-gradient RL to optimize flow policies can fail to shift the generation path toward the higher-value action mode a_1 (middle). VINE performs structured exploration in the noise space and constructs new denoising trajectories, enabling stable value-gradient propagation toward higher-value behavior (right).

1 INTRODUCTION

In robotic learning, generative control policies (GCPs) such as diffusion and flow-matching models have achieved remarkable success for behavior cloning in a wide range of robot manipulation and control tasks (Zhao et al., 2023; Chi et al., 2025; Black et al., 2024; Physical Intelligence et al.,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

2025). In general, GCPs produce executable actions through a multi-step generation process: starting from randomly sampled noise, each generation step takes the output from the previous step as input and uses a learned neural network to produce a progressively refined output, eventually yielding an executable action. Such iterative refinement processes are widely regarded as capable of representing complex and multimodal action distributions, offering greater expressive power than conventional policy parameterizations (Chi et al., 2025; Physical Intelligence et al., 2025).

To further enhance their capabilities, recent works have introduced reinforcement learning (RL) to GCPs (Chen et al., 2025; Zhang et al., 2025b; Yang et al., 2026; 2025a;b). However, existing studies have shown that directly optimizing GCPs with RL can lead to severe training instability (Zhang et al., 2026; Li & Levine, 2026; Park et al., 2025b; Psenka et al., 2024). Actor-critic RL frameworks use a learned value function to estimate future returns. The value function provides the optimization signal for improving the policy. Within this framework, a GCP serves as the policy and is trained to maximize expected returns. When a GCP is optimized directly using critic action gradients, those gradients pass through the full generation chain because the critic evaluates the final action. This optimization uses backpropagation through time (BPTT) and can suffer from poor performance and training instability. To improve stability, existing methods largely avoid propagating value gradients through the full denoising trajectory (Xu et al., 2026b). Specifically, prior methods typically adopt one of three strategies: (1) steering the generative policy through external guidance or auxiliary denoising objectives (Psenka et al., 2024; Wagenmaker et al., 2025); (2) moving RL updates to a Gaussian actor or a distilled one-step policy (Park et al., 2025b; Dong et al., 2025); and (3) using scalar value estimates to weight training objectives or select actions without the critic's action gradient (Peng et al., 2020; Hansen-Estruch et al., 2023). These approaches improve training stability but make different trade-offs. Strategy (1) uses guidance or auxiliary objectives instead of differentiating the final critic value through the full sampler. Strategy (2) optimizes a simpler actor rather than the full iterative generator, limiting how RL uses successive action refinement (Pan et al., 2025). Strategy (3) omits the critic's action gradient, which provides a direct, action-dependent signal toward higher-value behavior (Park et al., 2025b; Xu et al., 2026a). Previous empirical studies have shown that flow-based policies often underperform traditional Gaussian-based policies in reinforcement learning (Park et al., 2025a), suggesting that existing approaches struggle to fully exploit their high expressiveness and iterative generation while performing value-gradient optimization.

In this paper, we claim that stable RL for GCPs should not require sacrificing either iterative generation or value-gradient optimization. We therefore examine how the denoising process can adapt to changing value signals. In imitation learning, both the training objective and the target action distribution are fixed. The denoising process therefore only needs to progressively transform noise into an action sampled from this target distribution. But RL is inherently dynamic. The value function is updated throughout training, changing how existing behaviors are evaluated, while online policy rollouts may discover previously unseen actions with higher rewards. These changing value signals motivate sampling strategies that can move beyond existing action modes and adapt toward higher-value behavior. We investigate this challenge from the perspective of sampler design. This leads us to propose value-gradient iterative noise exploration VINE, a novel RL-oriented sampling strategy for generative control policies. VINE exploits a key property learned during flow matching training: diverse noise realizations are transported to different modes of the multimodal action distribution. By reconstructing intermediate states with fresh Gaussian noise, VINE explores alternative denoising paths without additional velocity-network evaluations. These trajectories provide more structured exploration across action modes while reducing the reliance on a single generation path. VINE yields more stable RL training while preserving iterative generation in simulated benchmarks and real-world robot learning.

2 Preliminaries

2.1 Reinforcement Learning and Actor-Critic Training

We consider a Markov Decision Process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, \gamma, R, \mu)$, where $\mathcal{S}$ is the state space, $\mathcal{A} = \mathbb{R}^d$ is the continuous action space, $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition function, $\gamma \in [0, 1)$ is the discount factor, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\mu \in \Delta(\mathcal{S})$ is the initial state distribution. We denote by $\mathcal{D} = \{(s_i, a_i, r_i, s'_i)\}_{i=1}^{|\mathcal{D}|}$ the data buffer used for actor-critic training. In offline RL, $\mathcal{D}$ is fixed and pre-collected; in online RL, $\mathcal{D}$ is a replay buffer that is continually expanded

through environment interaction. The goal is to learn a policy $\pi_\theta : \mathcal{S} \to \Delta(\mathcal{A})$ that maximizes the expected discounted return $J(\pi_\theta) = \mathbb{E}\left[\sum_{\ell=0}^{\infty} \gamma^\ell R(s_\ell, a_\ell)\right]$, where trajectories follow $s_0 \sim \mu$, $a_\ell \sim \pi_\theta(\cdot \mid s_\ell)$, and $s_{\ell+1} \sim P(\cdot \mid s_\ell, a_\ell)$. To optimize this objective from sampled transitions in $\mathcal{D}$, we consider a behavior-regularized actor-critic framework that improves policy actions using a learned critic while penalizing deviations from dataset actions (Wu et al., 2019; Fujimoto & Gu, 2021; Tarasov et al., 2023). A simplified instantiation uses the following critic and actor objectives:

$$\mathcal{L}_{\text{critic}}(\phi) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\left[\left(Q_\phi(s,a) - r - \gamma Q_{\bar{\phi}}(s', \pi_\theta(s'))\right)^2\right], \tag{1}$$

$$\mathcal{L}_{\text{actor}}(\theta) = \mathbb{E}_{(s,a)\sim\mathcal{D}}\left[-Q_\phi(s, \pi_\theta(s)) + \alpha \left\|a - \pi_\theta(s)\right\|_2^2\right], \tag{2}$$

Here, Q_ϕ approximates the action-value function, $Q_{\bar{\phi}}$ is a target critic, and $\alpha \geq 0$ controls the strength of behavior regularization. The bootstrap term is omitted for terminal transitions. The actor objective $\mathcal{L}_{\text{actor}}$ increases the critic-estimated value of generated actions while penalizing their deviation from dataset actions. With the critic held fixed during the actor update, the value term in $\mathcal{L}_{\text{actor}}$ is optimized by differentiating through the policy π_θ. For one-step policies, this derivative passes through a single action-generation stage. For multi-step generative control policies, direct differentiation of the same objective propagates the critic's action gradient through the multi-step generation chain. We next make this dependence explicit.

2.2 Generative Control Policies

Generative control policies model conditional action distributions by transforming base noise into actions through a sequence of state-conditioned sampling transitions. Let $x_k \in \mathbb{R}^d$ denote the intermediate sampling state at step k, distinct from the environment state s. For an integer number of steps $K \geq 1$, the sampling transitions take the general form

$$x_{k+1} = \mathcal{T}_{k,\theta}(x_k;\, s, z_k), \qquad k = 0, \ldots, K-1, \tag{3}$$

where x_0 is sampled from an easily sampled base distribution, z_k denotes possible sampling noise, and the final state x_K is executed as the action a. Flow matching (Lipman et al., 2023; Liu et al., 2023b; Albergo & Vanden-Eijnden, 2023) and diffusion models (Ho et al., 2020; Song et al., 2021) are two representative constructions of generative control policies. Although their parameterizations and sampling transitions differ, both methods transform noise into actions through a sequence of intermediate denoising or transport states. Specifically, flow matching represents the generation process as a continuous-time transport driven by a state-conditioned velocity field $v_\theta(x, t;\, s) \in \mathbb{R}^d$. With I_d denoting the $d \times d$ identity matrix, the continuous-time dynamics are described by the ordinary differential equation (ODE)

$$\mathrm{d}\hat{X}_t = v_\theta(\hat{X}_t, t;\, s)\,\mathrm{d}t, \qquad \hat{X}_0 \sim p_0, \qquad p_0 = \mathcal{N}(0, I_d), \tag{4}$$

where $\hat{X}_t$ denotes the intermediate state of the generative process at time $t \in [0, 1]$. The velocity field $v_\theta(\hat{X}_t, t;\, s)$ specifies the instantaneous direction and speed of this transport process. For the conditional flow-matching construction, we independently sample a state-action pair $(s, a) \sim \mathcal{D}$, a time $t \sim \mathcal{U}[0, 1]$, and Gaussian noise $z \sim \mathcal{N}(0, I_d)$. The interpolation state is $x_t = ta + (1-t)z$, and the velocity field is trained to regress the target velocity $a - z$:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{(s,a)\sim\mathcal{D},\, t\sim\mathcal{U}[0,1],\, z\sim\mathcal{N}(0,I_d)}\left[\left\|v_\theta(x_t, t;\, s) - (a - z)\right\|_2^2\right]. \tag{5}$$

At inference, the continuous-time ODE is discretized into a sequence of Euler steps. Given a time grid $0 = t_0 < t_1 < \cdots < t_K = 1$ and an initial sample $x_0 \sim \mathcal{N}(0, I_d)$, the sampler recursively computes

$$x_{k+1} = x_k + (t_{k+1} - t_k) \cdot v_\theta(x_k, t_k;\, s), \qquad k = 0, \ldots, K-1, \qquad a = x_K. \tag{6}$$

For a uniform grid, $t_k = k/K$. The sampler requires one velocity-network evaluation per step, for a total of K evaluations. We use flow-matching notation in the following analysis; the general transition formulation in Eq. (3) also accommodates other differentiable generative samplers (Song et al., 2021; Kingma & Gao, 2023). Let $J_k \coloneqq \partial x_{k+1}/\partial x_k$. For a fixed state, critic, and realization of parameter-independent sampling randomness, the value gradient can be written as

$$\nabla_\theta Q_\phi(s, \pi_\theta(s)) = \sum_{j=0}^{K-1} \left(\left.\frac{\partial x_{j+1}}{\partial \theta}\right|_{x_j}\right)^\top \left(J_{j+1}^\top J_{j+2}^\top \cdots J_{K-1}^\top\right) \nabla_a Q_\phi(s, a)|_{a=x_K}. \tag{7}$$

Algorithm 1 VINE: Actor–Critic Training.

function GENERATE(s)
  $\hat{a}_0 \sim \mathcal{N}(0, I_d)$
  for $k = 0, \dots, K-1$ **do**
    $t_k \leftarrow k/K$
    ~~$x_{k+1} \leftarrow x_k + \frac{1}{K} v_\theta(x_k, t_k;\, s)$~~ ▷ Euler update
    $z_k \sim \mathcal{N}(0, I_d)$ ▷ VINE update
    $x_k \leftarrow t_k \hat{a}_k + (1 - t_k) z_k$
    $\hat{a}_{k+1} \leftarrow x_k + (1 - t_k) v_\theta(x_k, t_k;\, s)$
  end for
  return $\hat{a}_K$
end function
while not converged **do**
  Collect transitions with GENERATE(s) and add them to $\mathcal{D}$ ▷ Optionally for online RL
  Sample batch $\{(s, a, r, s')\} \sim \mathcal{D}$
  $a' \leftarrow$ GENERATE(s')
  Update ϕ to minimize $\mathbb{E}[(Q_\phi(s, a) - r - \gamma Q_{\bar{\phi}}(s', a'))^2]$ ▷ Critic update
  $\hat{a}_K \leftarrow$ GENERATE(s)
  Update θ to minimize $-Q_\phi(s, \hat{a}_K) + \alpha \|\hat{a}_K - a\|_2^2$ ▷ Train v_θ via Full-chain BPTT
end while
return $\pi_\theta(s) \equiv$ GENERATE(s)

The critic's action gradient propagates through products of the single-step Jacobians $\partial x_{k+1}/\partial x_k$ before contributing to the policy parameters at each generation step. Changing the sampler changes these Jacobians and therefore the propagation of value gradients. Products are ordered with increasing indices from left to right, with an empty product equal to identity matrix I_d. The full derivation is provided in Appendix B.

3 VINE: VALUE-GRADIENT ITERATIVE NOISE EXPLORATION

As discussed above, the value gradient in a generative control policy depends not only on the velocity network but also on the sampling dynamics that transport intermediate states to the final action. We now introduce VINE by modifying the sampling transitions of a flow-matching policy. The core idea is to reconstruct intermediate sampling states using fresh independent Gaussian noise from the same base distribution as in flow-matching training. Starting from the flow-matching transport process, the Euler update in Eq. (6) can be algebraically rewritten as

$$x_{k+1} = x_k + (t_{k+1} - t_k) v_\theta(x_k, t_k; s) \tag{8}$$
$$= t_{k+1} \left[x_k + (1 - t_k) v_\theta \right] + (1 - t_{k+1}) \left[x_k - t_k v_\theta \right] \tag{9}$$
$$= t_{k+1} \hat{a}_{k+1} + (1 - t_{k+1}) \hat{z}_{k+1}, \qquad k = 0, \dots, K-1, \tag{10}$$

where we define

$$\hat{a}_{k+1} \coloneqq x_k + (1 - t_k) v_\theta(x_k, t_k; s), \qquad \hat{z}_{k+1} \coloneqq x_k - t_k v_\theta(x_k, t_k; s). \tag{11}$$

This rewriting exposes the connection between Euler sampling and the flow-matching training construction. During training with Eq. (5), the input to the velocity model is constructed with interpolation states $x_t = ta + (1 - t)z$. At inference, the Euler update therefore estimates this local interpolation with $\hat{a}_{k+1}$ and $\hat{z}_{k+1}$, while using the learned velocity to transport x_k to x_{k+1}. These quantities are action and noise endpoint estimates induced by the velocity field. Motivated by this structure, VINE retains the action-endpoint estimate and replaces the inferred noise endpoint $\hat{z}_{k+1}$ with a fresh independent Gaussian sample z_{k+1}. Writing $\hat{a}_k$ for the endpoint estimate entering step k, VINE reconstructs the sampling state as

$$z_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_d), \qquad x_k = t_k \hat{a}_k + (1 - t_k) z_k. \tag{12}$$

Intuitively, this update re-noises the current action estimate, introducing stochastic variation into the intermediate sampling path. The velocity field is then used to predict the next endpoint:

$$\hat{a}_{k+1} = x_k + (1 - t_k) v_\theta(x_k, t_k; s), \qquad k = 0, \dots, K-1. \tag{13}$$

The full generation process of VINE generates an action over K sampling steps. At each step $k = 0, \dots, K-1$, it forms a sampling state using Eq. (12), predicts an endpoint using Eq. (13), and uses the resulting transition to obtain the next state. It returns $a = \hat{a}_K$. We set $z_0 = \hat{a}_0 \sim \mathcal{N}(0, I_d)$. The initial sampling state is $x_0 = z_0$ and does not depend on $\hat{a}_0$. Because $t_K = 1$, the final reconstruction satisfies $x_K = \hat{a}_K$; thus, no additional network evaluation is required. VINE draws a fresh noise variable z_k. Consequently, the intermediate states $\{x_k\}_{k=0}^{K-1}$ depend on the fresh noise sequence, rather than being determined solely by the initial noise sample. Furthermore, this construction changes how value gradient credits are assigned across the denoising chain. To make it explicit, the local state-transition Jacobians are

$$\text{Euler transition:} \qquad \partial x_{k+1}/\partial x_k = I_d + (t_{k+1} - t_k)\partial v_\theta(x_k, t_k; s)/\partial x_k, \tag{14}$$

$$\text{VINE transition:} \qquad \partial x_{k+1}/\partial x_k = t_{k+1}\big[I_d + (1 - t_k)\partial v_\theta(x_k, t_k; s)/\partial x_k\big]. \tag{15}$$

For Euler, the direct identity contribution has coefficient one at every transition. For VINE, this coefficient is t_{k+1}, which is strictly smaller than one for the interior transitions $k = 0, \dots, K-2$, while it equals one at the final transition. Unlike detaching the inferred noise endpoint in Euler sampling, which preserves the forward sampling rule but generally yields a surrogate gradient, VINE replaces that endpoint with independent Gaussian noise in the forward process. Consequently, its modified backward rule is the exact pathwise derivative of its own sampler. Complete derivations are provided in Appendix B. Section 5 empirically examines how this modified gradient geometry affects gradient transport and optimization behavior. We instantiate VINE in the actor-critic framework defined by Eqs. (1) and (2). VINE generates actions for critic targets, actor updates, and online environment interaction. During actor updates, gradients are propagated through all sampling transitions without stopping gradients at intermediate endpoint predictions. The complete procedure is presented in Algorithm 1.

4 Related Work

Policy representations for efficient RL. Gaussian and deterministic actors simplify critic-gradient optimization (Silver et al., 2014; Fujimoto & Gu, 2021). Efficient generative policies use consistency models (Ding & Jin, 2024), diffusion-regularized one-step actors (Chen et al., 2024), shortcut models (Espinosa-Dice et al., 2025), or distilled flow policies (Park et al., 2025b; Shin et al., 2026). These approaches can retain multimodal expressivity while reducing computation. VINE focuses on the complementary problem of optimizing an iterative policy through its sampling transitions.

RL optimization of diffusion and flow policies. We distinguish methods by whether critic action gradients traverse the full multi-step sampler. BPTT-free approaches use value-weighted losses (Zhang et al., 2025a; Ding et al., 2024), policy-gradient objectives (Ren et al., 2025; Zhang et al., 2025b; McAllister et al., 2026), single-step action reconstruction (Kang et al., 2023; Zhang et al., 2024), sampling guidance (Lu et al., 2023; Frans et al., 2025), action selection (Hansen-Estruch et al., 2023; Dong et al., 2025), or auxiliary policy adaptation (Yuan et al., 2024; Wagenmaker et al., 2025; Mark et al., 2025). Critic gradients can also enter denoising objectives (Psenka et al., 2024; Fang et al., 2025). QAM builds on adjoint matching (i Enrich et al., 2025; Li & Levine, 2026), while FlowDPG and DFQL use critic-gradient surrogates (Shi et al., 2026; Cao et al., 2026); these methods retain iterative generation without full-chain value-gradient BPTT. Value-Gradient BPTT methods directly differentiate the critic objective through the sampler (Wang et al., 2023; Zhang et al., 2026). VINE intervenes in the sampling dynamics while preserving the vanilla velocity parameterization, full value-gradient BPTT, and the number of network evaluations.

5 Experiments

We first visualize sampling under behavior cloning and RL, then evaluate offline RL on OGBench, online adaptation of a pretrained vision-language-action (VLA) model, and human-in-the-loop RL on a real robot. Ablations examine the sampling design. Implementation details are in Appendix C.

Toy example. We fit diffusion and flow-matching models to a two-dimensional five-mode Gaussian mixture, shown as gray points in Figure 2. We sample the diffusion model with DDPM and the flow model with Euler or VINE, recording trajectories and endpoints for multiple noise realizations. All three recover the five modes. DDPM has broader intermediate-state coverage but less

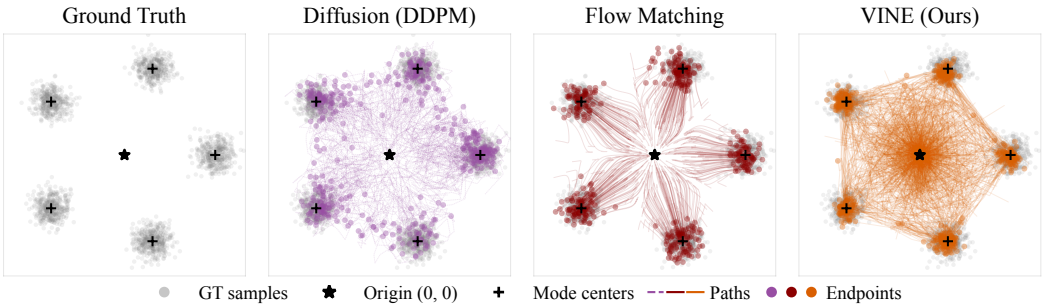

Figure 2: **Simulating a Probability Path with Different Samplers.** Ground-truth Gaussian-mixture samples are shown as gray points. We sample multiple initial noise realizations and visualize the corresponding intermediate denoising trajectories and estimated endpoints. VINE generates diverse denoising trajectories while retaining coverage of the target modes.

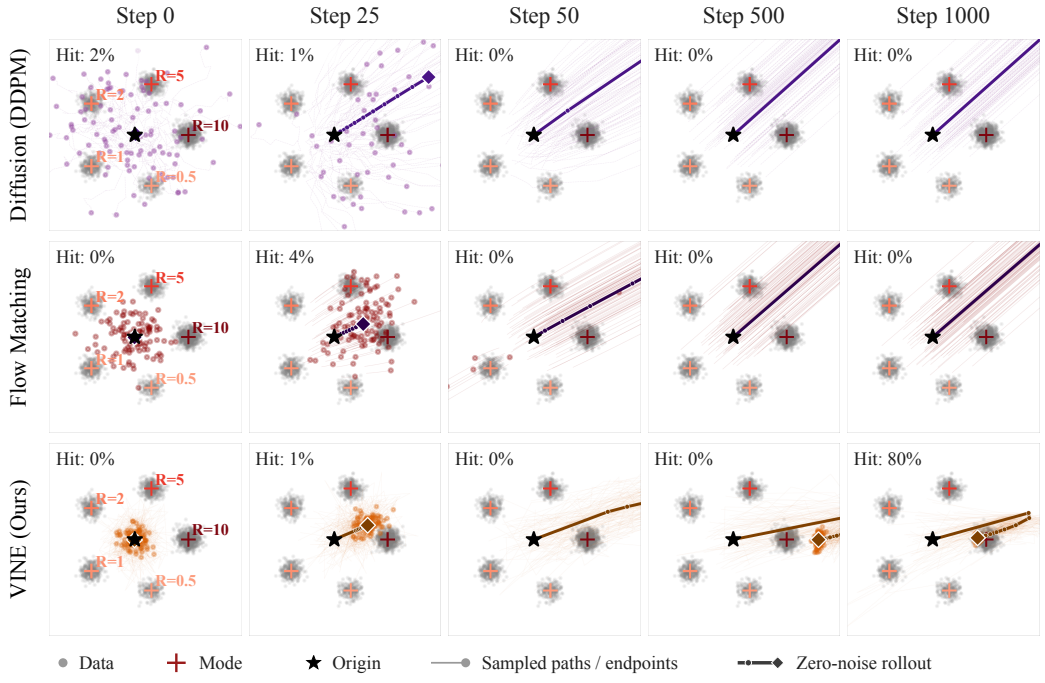

Figure 3: **Visualization of offline RL fine-tuning in a two-dimensional five-mode action space**. DDPM and Euler flow matching produce increasingly large off-target actions, whereas VINE progressively concentrates on the high-reward mode. Hit denotes the percentage of 100 sampled endpoints within distance 0.6 of the highest-reward mode. All panels use identical axis limits; mode rewards are labeled in the first column.

concentrated endpoints; Euler produces concentrated endpoints with isolated, petal-like trajectories. VINE combines concentrated endpoints with broader intermediate-state coverage. Furthermore, we assign different rewards to these modes to study RL adaptation with a learned value function. Figure 3 shows checkpoints at updates 0, 25, 50, 500, and 1000. DDPM and Euler initially approach the highest-reward mode but subsequently produce increasingly large action predictions. VINE maintains concentrated endpoints while shifting toward the highest-reward mode over the displayed training period. We also evaluate direct oracle-reward optimization without a learned critic; the protocol and results are provided in Appendix C.2.

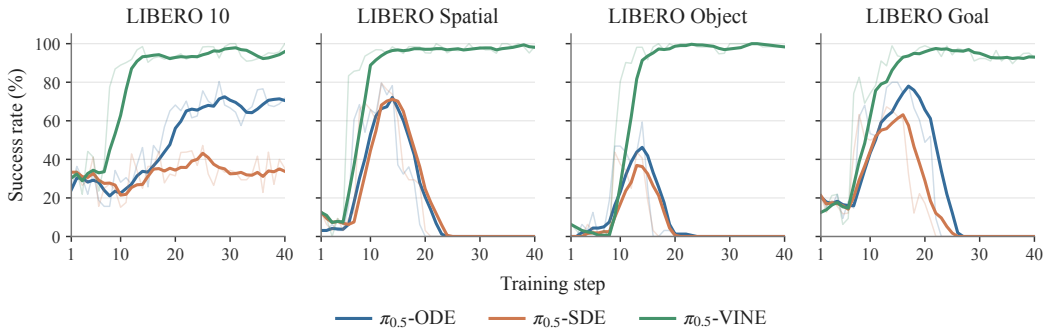

Figure 4: **Online RL adaptation of pretrained $\pi_{0.5}$ on LIBERO.** VINE achieves higher success rates and more consistent improvement than the baselines in the evaluated settings.

	Gaussian	BPTT-free											Value-Gradient BPTT		
Task Category	ReBRAC	FQL	FAWAC	IFQL	QAM	DAC	QSM	CGQL	CGQL-M	CGQL-L	DSRL	FEdit	FBRAC	BAM	VINE
antmaze-large	94	76	17	36	81	88	91	76	71	65	61	58	2	84	**99**
antmaze-giant	57	0	0	1	18	16	15	0	4	3	3	2	0	1	**76**
humanoidmaze-medium	69	68	24	86	67	83	83	60	42	62	53	22	39	60	**87**
humanoidmaze-large	17	9	0	24	11	0	10	5	6	6	3	3	0	5	**46**
scene-sparse	65	78	38	84	97	68	86	38	74	88	**99**	62	50	98	73
puzzle-3x3-sparse	79	70	3	**100**	**100**	68	53	48	**100**	90	87	99	0	56	95
puzzle-4x4-100M-sparse	0	5	0	0	0	0	0	24	0	0	0	**34**	15	0	28
cube-double	9	46	2	11	64	35	56	38	41	45	**74**	40	0	47	35
cube-triple	1	3	0	0	3	5	3	**8**	**8**	**8**	1	2	0	3	6
cube-quadruple-100M	9	2	0	2	3	3	19	0	1	0	2	5	0	0	**47**
all (50 tasks)	40	36	8	34	44	36	42	30	35	37	38	33	11	35	**59**

Table 1: **Offline RL results on OGBench.** BPTT-free methods update the generative control policy without backpropagating value gradients through the full sampling chain, while BPTT-based methods directly backpropagate value gradients through the sampling chain. VINE achieves competitive performance while retaining full-chain value-gradient BPTT.

Offline RL on OGBench. We evaluate VINE in the offline RL setting on 50 tasks across 10 domains from OGBench, covering a range of long-horizon, sparse-reward tasks in locomotion and manipulation, including antmaze-large, antmaze-giant, humanoidmaze-medium, humanoidmaze-large, scene-sparse, puzzle-3x3-sparse, puzzle-4x4-100M-sparse, cube-double, cube-triple, and cube-quadruple-100M. We compare VINE against representative Gaussian, flow-matching, and diffusion-based policy methods, grouped according to how they propagate value-gradient updates through or around the sampling chain, as discussed in Section 4. These include ReBRAC (Tarasov et al., 2023), FQL and FAWAC (Park et al., 2025b), IFQL (Hansen-Estruch et al., 2023), QAM (Li & Levine, 2026), DAC (Fang et al., 2025), QSM (Psenka et al., 2024), CGQL variants (Lu et al., 2023), DSRL (Wagenmaker et al., 2025), FEdit (Dong et al., 2025), FBRAC (Park et al., 2025b), and BAM (i Enrich et al., 2025; Li & Levine, 2026). Following the evaluation protocols, we train each task for 1M gradient updates with 12 random seeds and report the mean performance with 95% confidence intervals. As shown in Table 1, VINE outperforms BPTT-based methods such as FBRAC and BAM in aggregate. VINE achieves a comparable aggregate score to methods that avoid backpropagating through the sampling chain. These results demonstrate that VINE can stably optimize expressive generative policies with value gradients in the offline RL setting.

VINE on VLA for Multi-Task Offline-Online RL. We evaluate VINE on the 3B-parameter $\pi_{0.5}$ VLA (Intelligence et al., 2025) through online multi-task fine-tuning on LIBERO (Liu et al., 2023a). The policy consists of a vision-language backbone and a flow-matching action head for generating executable robot actions. To isolate the effect of the sampling procedure, all methods share the same $\pi_{0.5}$ initialization and differ only in their sampling procedures. We denote the default deterministic ODE sampler as $\pi_{0.5}$-ODE, the stochastic flow-matching sampler that injects noise throughout the generation process as $\pi_{0.5}$-SDE, and our noise-reconstructive sampler as $\pi_{0.5}$-VINE. Figure 4 shows the online learning curves, where each subplot reports the success rate averaged over the ten

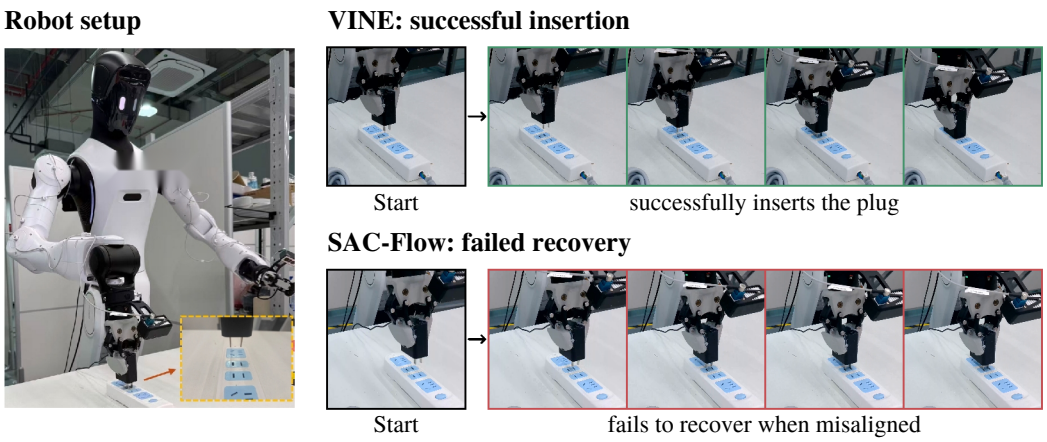

Figure 5: **Real-world setup and representative rollouts.** The left panel shows the robot and plug-insertion setup. The upper sequence shows a successful VINE insertion; the lower sequence shows a SAC-Flow rollout that fails to recover from misalignment. These selected examples illustrate behavior, while Table 2 reports success over 20 trials per method.

Table 2: **Real-world online RL results on a plug insertion task.** We train a robot to perform plug insertion in the real world using different RL algorithms. We report the success rate over 20 trials, online RL fine-tuning time, and the fraction of online rollout steps requiring human intervention.

Plug Insertion	BC init.	DSRL	RLT	EXPO	SAC-Flow	HIL-SERL	VINE
Success rate (↑)	10/20	17/20	17/20	19/20	12/20	16/20	**20/20**
Time (↓)	–	50 min	50 min	50 min	20 min	20 min	**20 min**
Human intervention (↓)	–	–	29.7%	56.3%	74.6%	55.9%	**19.2%**

tasks in the corresponding LIBERO suite. In the evaluated setting, both $\pi_{0.5}$-ODE and $\pi_{0.5}$-SDE exhibit unstable performance during multi-task fine-tuning, whereas $\pi_{0.5}$-VINE achieves more consistent policy improvement throughout training. These results demonstrate that VINE scales to a 3B-parameter VLA model and enables more stable multi-task online RL fine-tuning.

VINE on Real-World Online RL. We evaluate VINE on a real-world plug insertion task to assess whether stable value-gradient optimization transfers to physical robotic manipulation. As shown in Figure 5, the robot must pick up a plug from the table and insert it into a fixed socket, requiring precise contact-rich control under tight insertion tolerances. We compare VINE against representative real-world RL methods, including SAC-Flow (Zhang et al., 2026), HIL-SERL (Luo et al., 2025), DSRL (Wagenmaker et al., 2025), RLT (Xu et al., 2026a), and EXPO (Dong et al., 2025), in a human-in-the-loop RL setting (Luo et al., 2025; Xu et al., 2026a), where a human operator can intervene during autonomous execution and provide corrective actions. All methods take images and proprioceptive states as input and directly predict end-effector actions. We evaluate each method over 20 trials and report the success rate, wall-clock fine-tuning time, and human-intervention ratio in Table 2. VINE achieves 20/20 successful trials after 20 minutes of fine-tuning, with the lowest reported human-intervention ratio of 19.2%, demonstrating effective real-world online RL.

Ablation Study. To examine whether VINE's stochastic sampling procedure supports iterative action refinement, we evaluate the intermediate endpoint prediction $\hat{a}_k$, defined as the action predicted after k refinement steps. At each environment step, we execute $\hat{a}_k$ for $k \in \{1, 6, 10\}$. Figure 7 shows the resulting trajectories on AntMaze-giant. VINE's additional refinement steps improve closed-loop control performance. We compare uniform, two shifted, and Beta-quantile inference time grids without retraining, retaining ten evaluations and identical time endpoints (Appendix C.5). To examine the role of noise resampling during training, we compare independent Gaussian noise at each nonzero refinement step with a shared noise sample reused across steps within each actor update. Figure 6(b) reports task performance. Independent resampling achieves higher mean success across the evaluated domains. To compare gradient variation during training, we plot normalized gradient-norm traces for VINE and Euler flow matching trained with BPTT on AntMaze-giant. Fig-

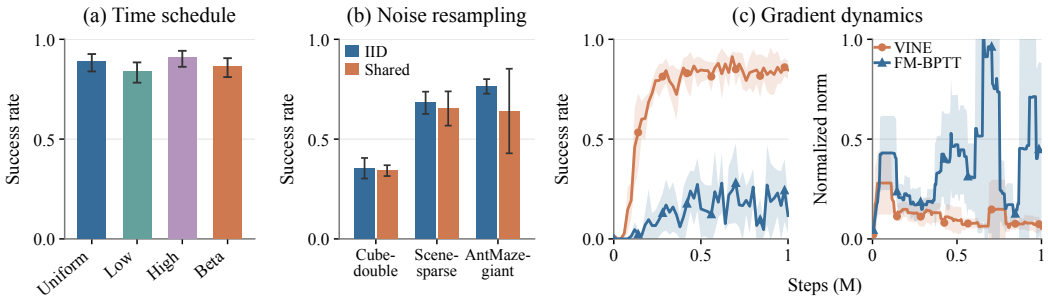

Figure 6: **Ablation study.** (a) Inference success under different time schedules $\{t_k\}$ using the same checkpoint. (b) Independent noise z_k resampling at each refinement step versus reusing a shared noise sample. (c) Success rates and normalized gradient-norm traces during training on AntMaze-giant, comparing VINE with Euler using BPTT.

ure 6(c) shows smaller fluctuations in the normalized trace for VINE than for flow matching with Euler (FM-BPTT).

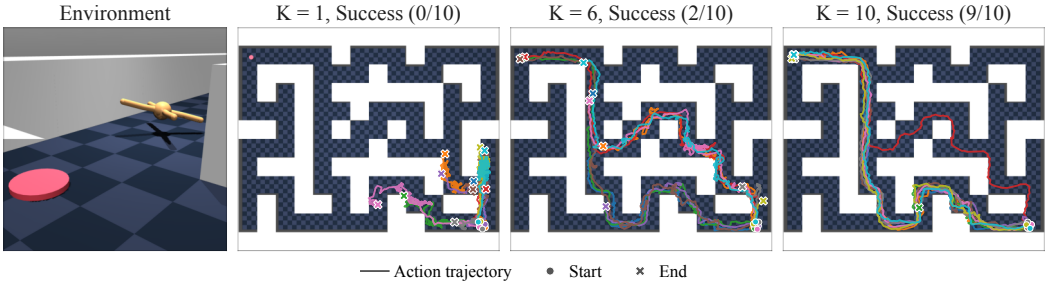

Figure 7: **Effect of intermediate action refinement.** Executed XY trajectories on AntMaze-giant using the first $K \in \{1, 6, 10\}$ steps of the same checkpoint. Success counts successful episodes out of ten rollouts per condition. Different colors indicate different trials.

6 Conclusion

We show that sampler design plays an important role in value-gradient RL for generative control policies. VINE reconstructs intermediate states with fresh Gaussian noise while retaining full-chain gradients and the same network-evaluation budget as Euler. Experiments demonstrate competitive offline RL on OGBench tasks, more stable online adaptation of a pretrained 3B-parameter VLA on LIBERO, and effective online learning for real-world plug insertion. Together, these results show that iterative generation can support end-to-end value-gradient optimization and highlight sampler design as a practical route to improving generative control policies. These results demonstrate that iterative generation is not inherently incompatible with value-gradient RL, opening a path toward fully leveraging generative policies for reinforcement learning.

AI USE STATEMENT

In this work, we used generative AI tools for grammar diagnosis and coding assistance. We did not use generative AI tools to generate research ideas, develop methods, design experiments, analyze results, or write the manuscript. All AI-assisted code was reviewed, verified, and tested by the authors for correctness. We have reviewed all AI-assisted work and take full responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Michael S. Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *International Conference on Learning Representations*, 2023.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Shicheng Cao, Jingrui Jia, Wenyu Li, Feng Duan, Tao Zhang, and Shengbo Li. Direct flow Q-learning. In *International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=RdkOaK4q6p.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Xiang Li, Quanlu Zhang, Zhaofei Yu, et al. $\pi_{\texttt{rl}}$: Online rl fine-tuning for flow-based vision-language-action models. *arXiv preprint*, arXiv:2510.25889, 2025. arXiv preprint arXiv:2510.25889.

Tianyu Chen, Zhendong Wang, and Mingyuan Zhou. Diffusion policies creating a trust region for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 2024.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10–11):1684–1704, 2025. doi: 10.1177/02783649241273668.

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. In *Neural Information Processing Systems*, 2024.

Zihan Ding and Chi Jin. Consistency models as a rich and efficient policy class for reinforcement learning. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 53047–53066, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/e9e1a0abc1a5b19a4aeb80dab19c82ae-Paper-Conference.pdf.

Perry Dong, Qiyang Li, Dorsa Sadigh, and Chelsea Finn. EXPO: Stable reinforcement learning with expressive policies. *arXiv preprint arXiv:2507.07986*, 2025.

Nicolas Espinosa-Dice, Yiyi Zhang, Yiding Chen, Bradley Guo, Owen Oertell, Gokul Swamy, Kiante Brantley, and Wen Sun. Scaling offline RL via efficient and expressive shortcut models. *arXiv preprint arXiv:2505.22866*, 2025.

Linjiajie Fang, Ruoxue Liu, Jing Zhang, Wenjia Wang, and Bingyi Jing. Diffusion actor-critic: Formulating constrained policy iteration as diffusion noise regression for offline reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

Kevin Frans, Seohong Park, Pieter Abbeel, and Sergey Levine. Diffusion guidance is a controllable policy improvement operator. *arXiv preprint arXiv:2505.23458*, 2025.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 2021.

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.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus), 2016. URL https://arxiv.org/abs/1606.08415.

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

Carles Domingo i Enrich, Michal Drozdzal, Brian Karrer, and Ricky T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=xQBRrtQM8u.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Bingyi Kang, Xiao Ma, Chao Du, Tianyu Pang, and Shuicheng Yan. Efficient diffusion policies for offline reinforcement learning. In *Neural Information Processing Systems*, 2023.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. URL https://arxiv.org/abs/1412.6980.

Diederik P. Kingma and Ruiqi Gao. Understanding diffusion objectives as the elbo with simple data augmentation. *Advances in Neural Information Processing Systems*, 2023.

Qiyang Li and Sergey Levine. Q-learning with adjoint matching. *International Conference on Learning Representations*, 2026.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *International Conference on Learning Representations*, 2023.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023a.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *International Conference on Learning Representations*, 2023b.

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*, 2023.

Jianlan Luo, Charles Xu, Jeffrey Wu, and Sergey Levine. Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning. *Science Robotics*, 10(105):eads5033, 2025.

Max Sobol Mark, Tian Gao, Georgia Gabriela Sampaio, Mohan Kumar Srirama, Archit Sharma, Chelsea Finn, and Aviral Kumar. Policy-agnostic RL: Offline RL and online RL fine-tuning of any class and backbone. In *Robot Learning Workshop*, 2025.

David McAllister, Songwei Ge, Brent Yi, Chung Min Kim, Ethan Weber, Hongsuk Choi, Haiwen Feng, and Angjoo Kanazawa. Flow matching policy gradients. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=eoEmoKoQpJ.

Chaoyi Pan, Giri Anantharaman, Nai-Chieh Huang, Claire Jin, Daniel Pfrommer, Chenyang Yuan, Frank Permenter, Guannan Qu, Nicholas Boffi, Guanya Shi, et al. Much ado about noising: Dispelling the myths of generative robotic control. *arXiv preprint arXiv:2512.01809*, 2025.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations*, volume 2025, pp. 94937–94982, 2025a.

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 48104–48127. PMLR, 13–19 Jul 2025b. URL https://proceedings.mlr.press/v267/park25f.html.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage weighted regression: Simple and scalable off-policy reinforcement learning, 2020. URL https://openreview.net/forum?id=H1gdF34FvS.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. $\pi^*_{0.6}$: a vla that learns from experience. *arXiv preprint*, arXiv:2511.14759, 2025. arXiv preprint arXiv:2511.14759.

Michael Psenka, Alejandro Escontrela, Pieter Abbeel, and Yi Ma. Learning a diffusion model policy from rewards via Q-score matching. In *International Conference on Machine Learning*, 2024.

Allen Z. Ren, Justin Lidard, Lars Lien Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=mEpqHvbD2h.

Kexin Shi, Junyao Shi, Poorvi Hebbar, Zhuolun Zhao, Tarun Amarnath, Yifan Su, Shikhar Bahl, and Deepak Pathak. Flowdpg: Deterministic policy gradient on flow matching policies for real-world manipulation, 2026. URL https://arxiv.org/abs/2606.22303.

Yongjae Shin, Jongseong Chae, Jongeui Park, and Youngchul Sung. Flow matching with injected noise for offline-to-online reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=6wd38R8L0Z.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In Eric P. Xing and Tony Jebara (eds.), *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pp. 387–395, Bejing, China, 22–24 Jun 2014. PMLR. URL https://proceedings.mlr.press/v32/silver14.html.

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

Denis Tarasov, Alexander Nikulin, Dmitry Akimov, Vladislav Kurenkov, and Sergey Kolesnikov. Corl: Research-oriented deep offline reinforcement learning library. *Advances in Neural Information Processing Systems*, 2023.

Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. *Conference on Robot Learning*, 2025.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *International Conference on Learning Representations*, 2023.

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

Charles Xu, Jost Tobias Springenberg, Michael Equi, Ali Amin, Adnan Esmail, Sergey Levine, and Liyiming Ke. Rl token: Bootstrapping online rl with vision-language-action models. *arXiv preprint arXiv:2604.23073*, 2026a.

Haoran Xu, Kaiwen Hu, Somayeh Sojoudi, and Amy Zhang. Reinforcement learning via value gradient flow. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=JLL4VNVhM9.

Rushuai Yang, Zhiyuan Feng, Tianxiang Zhang, Kaixin Wang, Chuheng Zhang, Li Zhao, Xiu Su, Yi Chen, and Jiang Bian. Discover, learn, and reinforce: Scaling vision-language-action pretraining with diverse RL-generated trajectories, 2025a. URL https://arxiv.org/abs/2511.19528.

Rushuai Yang, Hangxing Wei, Ran Zhang, Zhiyuan Feng, Xiaoyu Chen, Tong Li, Chuheng Zhang, Li Zhao, Jiang Bian, Xiu Su, and Yi Chen. Beyond human demonstrations: Diffusion-based reinforcement learning to generate data for VLA training, 2025b. URL https://arxiv.org/abs/2509.19752.

Rushuai Yang, Hecheng Wang, Zhichao Wu, Chiming Liu, Xiaohan Yan, Xuan Du, Shuoyu Yue, Chuheng Zhang, Yunlong Wang, Yongcheng Liu, Lizhe Qi, Yi Chen, Wei Shan, and Maoqing Yao. ALOE: Action-level off-policy evaluation for vision-language-action model post-training, 2026. URL https://arxiv.org/abs/2602.12691.

Xiu Yuan, Tongzhou Mu, Stone Tao, Yunhao Fang, Mengke Zhang, and Hao Su. Policy decorator: Model-agnostic online refinement for large policy model. *arXiv preprint arXiv:2412.13630*, 2024.

Hongzhi Zang, Mingjie Wei, Si Xu, Yongji Wu, Zhen Guo, Yuanqing Wang, Hao Lin, Liangzhi Shi, Yuqing Xie, Zhexuan Xu, et al. Rlinf-vla: A unified and efficient framework for vla+ rl training. *arXiv preprint arXiv:2510.06710*, 2025.

Ruoqi Zhang, Ziwei Luo, Jens Sjölund, Thomas B Schön, and Per Mattsson. Entropy-regularized diffusion policy with Q-ensembles for offline reinforcement learning. In *Neural Information Processing Systems*, 2024.

Shiyuan Zhang, Weitong Zhang, and Quanquan Gu. Energy-weighted flow matching for offline reinforcement learning. In *International Conference on Learning Representations*, 2025a.

Tonghe Zhang, Chao Yu, Sichang Su, and Yu Wang. Reinflow: Fine-tuning flow matching policy with online reinforcement learning. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 106282–106319. Curran Associates, Inc., 2025b. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/98d6f928497d9eaad9f320fb2db3040d-Paper-Conference.pdf.

Yixian Zhang, Shu'ang Yu, Tonghe Zhang, Mo Guang, Haojia Hui, Kaiwen Long, Yu Wang, Chao Yu, and Wenbo Ding. Sac flow: Sample-efficient reinforcement learning of flow-based policies via velocity-reparameterized sequential modeling, 2026. URL https://arxiv.org/abs/2509.25756.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems*, 2023.