Optimizing Diffusion-based model free Reinforcement Learning

Kaustabh Paul[1]

Abstract— **Diffusion-based reinforcement learning (RL) has emerged in recent years as a viable paradigm for continuous control by merging powerful denoising diffusion models with stable off-policy optimization. This study examines two prominent online, model-free, off-policy algorithms—DIffusion POlicy (DIPO) and Q-weighted Variational Policy Optimization (QVPO)—to evaluate the influence of various diffusion sampling strategies on learning dynamics. While prior work has primarily relied on the conventional Denoising Diffusion Probabilistic Model (DDPM) sampler, a range of alternative samplers is implemented and assessed for their impact on convergence rate, sample complexity, and overall policy effectiveness. These methods are further applied and fine-tuned on a set of low-dimensional simulated manipulation tasks, including pen spinning and single-object pick-and-place. Although these alternative strategies did not yield the anticipated performance improvements, the resulting analysis reveals critical insights into the limitations, hyperparameter sensitivities, and tuning challenges inherent to diffusion-based policies in contact-rich environments. The findings underscore the importance of sampler selection in diffusion-based RL and provide actionable guidance for the development of future simulation benchmarks.**

I. INTRODUCTION

Reinforcement learning (RL) algorithms have long been recognized as effective tools for optimal sequential decision-making, enabling agents to acquire policies that maximize cumulative reward through trial-and-error interactions in uncertain environments. Within the model-free paradigm, policies or value functions are optimized directly from experience without constructing an explicit model of environmental dynamics. Model-free methods are typically categorized into three classes: value-based approaches that learn action–value functions and derive policies via greedy selection, policy-based methods that adjust parameterized stochastic policies using gradient estimates, and actor-critic architectures that integrate both by having an actor propose actions and a critic evaluate them via value functions.

Contemporary RL research often emphasizes online, model-free, off-policy algorithms due to their superior sample efficiency. In online learning, the agent continuously interacts with the environment and updates its policy based on newly gathered experiences. Off-policy learning further enhances efficiency by leveraging a replay buffer of past interactions, which may be generated by different behavior policies. Actor-critic methods are widely adopted in this context, as the critic's value estimates help stabilize the high-variance updates inherent in policy gradients, albeit at the cost of increased hyperparameter sensitivity and more complex tuning requirements [3].

Despite their advantages, these methods remain susceptible to instability and insufficient exploration. The confluence of bootstrapping, function approximation, and off-policy updates—referred to as the "deadly triad"—can result in divergent or oscillatory value estimates [3]. Moreover, conventional Gaussian policy parameterizations constrain action distributions to a unimodal form, thereby limiting exploration, impeding convergence, and intensifying estimation bias [1].

To address these limitations, generative models have been incorporated into RL frameworks. Generative Adversarial Imitation Learning (GAIL) employs generative adversarial networks (GANs) to replicate expert behavior in offline settings but is prone to mode collapse and distribution shift when deployed online [5]. Variational autoencoders (VAEs) offer improved stability during offline training but often generate averaged ("blurry") outputs, rendering them inadequate for modeling the multimodal action distributions required in continuous control tasks [4]. Decision Transformers reinterpret RL as a sequence modeling problem using transformer architectures, achieving competitive results on offline benchmarks while suffering from high computational demands and low sample efficiency in online contexts [6].

A promising alternative involves representing policies using denoising diffusion probabilistic models. These models iteratively transform noise into structured, multimodal action samples through a learned reverse process, supporting stable likelihood-based training and enhanced representational flexibility. Two recent algorithms—DIffusion POlicy (DIPO) and Q-weighted Variational Policy Optimization (QVPO)—integrate diffusion models into the model-free, online, off-policy RL setting to address exploration inefficiencies and stability concerns characteristic of traditional approaches. The present analysis investigates the influence of different diffusion samplers on the learning dynamics, sample efficiency, and ultimate performance of DIPO and QVPO, and offers preliminary simulation-based insights into practical considerations for tuning these methods in manipulation tasks.

II. RELATED WORK

A. Diffusion Models for Policy Representation

Denoising diffusion probabilistic models have been adapted from generative modeling to RL by treating the policy as a conditional diffusion model. At each timestep, the denoiser network ϵ_θ is conditioned on the state s and

*This work was not supported by any organization

[1]Kaustabh Paul is with the Department of Electrical and Computer Engineering, Carnegie Mellon University, 5000 Forbes Avenue, PA 15213, USA kaustabp@andrew.cmu.edu

Code and additional materials available at https://github.com/RedTorus/diffRL

diffusion index t, and learns to map noisy action samples back toward high-likelihood regions of the action space.

B. Denoising Diffusion Probabilistic Models (DDPM)

The DDPM framework [7] defines a forward process that gradually corrupts data x_0 into noise:

$$q(x_t \mid x_{t-1}) = \mathcal{N}\big(x_t; \sqrt{1-\beta_t}\, x_{t-1},\, \beta_t I\big)$$

The cumulative product of noise coefficients is

$$\bar{\alpha}_t = \prod_{i=1}^{t} (1-\beta_i)$$

A learnable reverse (denoising) process is parameterized as

$$p_\theta(x_{t-1} \mid x_t) = \mathcal{N}\big(x_{t-1}; \mu_\theta(x_t,t),\, \Sigma_\theta(x_t,t)\big)$$

With the choice

$$\mu_\theta(x_t,t) = \frac{1}{\sqrt{1-\beta_t}}\Big(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\,\epsilon_\theta(x_t,t)\Big)$$

$$\Sigma_\theta(x_t,t) = \beta_t I$$

the variational bound on the negative log-likelihood simplifies to a mean-squared error on the noise prediction:

$$\mathcal{L}_{\text{simple}} = E_{x_0\sim p_{\text{data}},\,\epsilon\sim\mathcal{N}(0,I),\,t}\Big\|\epsilon - \epsilon_\theta\big(\sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\,\epsilon,\; t\big)\Big\|^2$$

Sampling proceeds via ancestral sampling. Initialize

$$x_T \sim \mathcal{N}(0, I)$$

then for each $t = T, \ldots, 1$:

$$x_{t-1} = \mu_\theta(x_t,t) + \sqrt{\beta_t}\, z_t$$

$$z_t \sim \mathcal{N}(0, I)$$

C. DIffusion POlicy (DIPO)

The DIPO algorithm [1] is the first instance of an online, off-policy actor-critic framework using diffusion policies. A standard replay buffer is paired with a *diffusion buffer* of noisy action trajectories. During actor updates, each noisy action a_t is first nudged by the critic's gradient:

$$\hat{a} = a + \alpha\, \nabla_a Q(s,a), \tag{1}$$

and then the denoiser is trained via a mean-squared error on the noise prediction,

$$\mathcal{L}_{\text{actor}}^{\text{DIPO}} = E_{(s,a),\,t,\,\epsilon}\big\|\epsilon - \epsilon_\theta(a_t,s,t)\big\|^2. \tag{2}$$

DIPO comes with a convergence guarantee under standard MDP assumptions, providing a theoretical foundation for its multimodal policy representation. Empirically, extensive experiments on MuJoCo continuous control benchmarks demonstrate that DIPO consistently outperforms strong baselines such as SAC, PPO, and TD3 in both cumulative reward and sample efficiency [1].

D. Q-weighted Variational Policy Optimization (QVPO)

Building on DIPO, QVPO [2] weights each denoising sample by a value-based factor

$$w\big(Q(s,a_0)\big) = \exp\big(Q(s,a_0)/\eta\big) \tag{3}$$

and incorporates entropy regularization into the diffusion process to further encourage exploration. The actor loss becomes

$$\mathcal{L}_{\text{actor}}^{\text{QVPO}} = E_{(s,a_0),\,t,\,\epsilon}\Big[w\big(Q(s,a_0)\big)\,\|\epsilon - \epsilon_\theta(a_t,s,t)\|^2\Big]. \tag{4}$$

This Q-weighted variational objective is shown to be a tight lower bound of the true policy objective, and its entropy term is specially designed to handle the intractable log-likelihood of diffusion policies. On standard MuJoCo tasks, QVPO achieves state-of-the-art performance, surpassing both DIPO and conventional off-policy methods in cumulative reward and sample efficiency [2].

E. Discussion

Both DIPO and QVPO replace the traditional bootstrapped actor–critic update

$$\nabla_\theta J \approx E_{(s,a)\sim\mathcal{D}}\big[\nabla_\theta \pi_\theta(s)\, \nabla_a Q(s,a)\big]$$

with a simple mean-squared error (MSE) regression on diffusion denoising targets. This removes actor bootstrapping and yields markedly more stable training dynamics.

Moreover, by sampling actions through a conditional diffusion model and training the denoiser to reconstruct complex noise patterns, these methods naturally capture rich, multimodal action distributions—overcoming the unimodal limitations of Gaussian policies.

Sample efficiency is further enhanced by guidance mechanisms: DIPO's Q-gradient relabeling produces higher-quality actor targets from the diffusion buffer, while QVPO's exponential weighting skews training toward high-value regions of the action space.

Both DIPO and QVPO employ the standard DDPM (discrete-time denoising diffusion probabilistic model) sampler for both forward noising and reverse denoising, without investigating alternative sampler designs (e.g. ODE-based solvers or SDE-based samplers) and their impact on convergence or final performance.

Finally, the experimental evaluations in these works are confined to benchmark continuous-control domains (e.g. MuJoCo locomotion suites) and simple simulated manipulation tasks. They do not explore more challenging, highly multimodal environments or contact-rich scenarios, leaving open questions about diffusion-policy robustness and sampler choice in complex settings.

III. Methodology

A. Diffusion Policy Training

The actor network is implemented as a conditional denoising diffusion model, trained to reconstruct clean actions

from noisy samples drawn during each update. In practice, transitions (s, a) are drawn from the replay buffer, noise is added according to the diffusion schedule, and the denoiser is optimized via a simple MSE loss. DIPO enhances this process by first adjusting noisy actions with a Q-gradient step, while QVPO applies a Q-dependent weighting to the same objective. Crucially, the choice of sampler—whether standard DDPM ancestral steps, faster DDIM or other ODE solvers, or SDE integrators like Heun or Predictor–Corrector—can significantly affect sample quality, training stability, and computational cost, suggesting that exploring alternative sampler designs may yield further performance gains.

B. Critic Network

The critic network $Q_\phi(s, a)$ is trained in an off-policy manner using temporal-difference learning:

$$\mathcal{L}_{\text{critic}} = E_{(s,a,s',a')\sim\mathcal{D}}\Big[r(s,a)+\gamma\, Q_{\phi^-}(s',\,\tilde{a}')-Q_\phi(s,a)\Big]^2,$$

where $\tilde{a}'$ is sampled from the current diffusion policy, and Q_{ϕ^-} denotes a target network. The critic provides both the value estimates and, in the case of DIPO, the action gradients $\nabla_a Q(s, a)$ used to guide diffusion sampling. A stable critic is thus essential: inaccuracies can degrade the quality of actor targets and undermine the benefits of the diffusion framework.

To keep the target network Q_{ϕ^-} closely aligned yet more stable than the online critic, its parameters are updated via a soft-update rule after each critic step:

$$\phi^- \;\leftarrow\; \tau\,\phi + (1-\tau)\,\phi^-,$$

where $\tau \ll 1$ controls the update rate. This slow-moving average helps prevent harmful feedback loops between the actor and critic during training.

By separating policy generation (via diffusion sampling and denoising) from value estimation (via a bootstrapped critic), this methodology sidesteps the high-variance, bootstrapped policy gradients of traditional actor-critic algorithms and leverages the flexibility of modern sampler designs to balance sample quality against computational efficiency.

C. SDE-Based Samplers

Stochastic samplers simulate the reverse-time SDE directly, interleaving deterministic drift steps with random noise injections. By approximating the score function with the diffusion model's denoiser, these methods generate high-quality samples that closely track the true continuous-time dynamics, at the expense of requiring careful step-size control and multiple noise draws.

$$\mathrm{d}x = g(x,t)\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{w}_t,$$

where in practice

$$g(x,t) \approx -\frac{1}{2}\,\beta(t)\,x \;+\; \frac{\beta(t)}{\sqrt{1-\bar{\alpha}_t}}\,\epsilon_\theta(x,t).$$

These samplers often yield faithful reconstructions of the diffusion trajectory, but their random fluctuations can require many fine-grained steps:

- **Euler–Maruyama (DDPM):** One drift evaluation and one noise draw per step:

  $$x_{t-1} = x_t + g(x_t,t)\,\Delta t + \sqrt{\beta_t\,\Delta t}\;z, \quad z\sim\mathcal{N}(0,I).$$

  This baseline sampler is straightforward to implement and robust for small step sizes, but its first-order accuracy (local bias $O(\Delta t^2)$) means that decreasing errors demands many tiny steps [7].

- **Heun's Method:** Two drift evaluations and one noise draw per step. Predictor:

  $$x^* = x_t + g(x_t,t)\,\Delta t + \sqrt{\beta_t\,\Delta t}\;z.$$

  Corrector:

  $$x_{t-1} = x_t + \frac{1}{2}\big(g(x_t,t)+g(x^*,t-\Delta t)\big)\,\Delta t + \sqrt{\beta_t\,\Delta t}\;z.$$

  By effectively averaging the drift over the interval, Heun's method achieves second-order accuracy (local bias $O(\Delta t^3)$), allowing larger steps with similar fidelity [8].

- **Predictor–Corrector (PC):** Combines small Langevin corrections with an ancestral DDPM step. Corrector (for $j = 0, \ldots, C-1$):

  $$x_t^{(j+1)} = x_t^{(j)} + \varepsilon_{\text{corr}}\,g\big(x_t^{(j)},t\big) + \sqrt{2\,\varepsilon_{\text{corr}}}\;z_j, \quad z_j\sim\mathcal{N}(0,I).$$

  Predictor:

  $$x_{t-1} = x_t^{(C)} + g\big(x_t^{(C)},t\big)\,\Delta t + \sqrt{\beta_t\,\Delta t}\;z, \quad z\sim\mathcal{N}(0,I).$$

  Interleaving Langevin-style corrections before each ancestral update yields notably sharper samples, but introduces stochasticity that can complicate stability if C or $\varepsilon_{\text{corr}}$ are not well tuned [9].

D. ODE-Based Samplers

Deterministic samplers integrate the probability-flow ODE instead of simulating noise, providing samples that exactly match the diffusion marginals with fewer function calls. This approach removes random fluctuations and can converge in a handful of large steps, though it may accumulate systematic bias if the integrator is not sufficiently accurate.

The continuous-time form is obtained by replacing noise with a continuity term:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -\frac{1}{2}\,\beta(t)\,x \;-\; \beta(t)\,\nabla_x \log p_t(x),$$

or in "(σ)-space":

$$\frac{\mathrm{d}x}{\mathrm{d}\sigma} = \frac{x-\hat{x}_0(x,t)}{\sigma}, \quad \hat{x}_0(x,t) = \frac{x-\sqrt{1-\bar{\alpha}_t}\,\epsilon_\theta(x,t)}{\sqrt{\bar{\alpha}_t}}.$$

These ODE solvers can traverse the noise-scale dimension in large increments:

- **DDIM:** A single Euler step per sample:

$$h_i = \sigma_{i+1} - \sigma_i, \quad x_{i+1} = x_i + h_i \, \frac{x_i - \hat{x}_0(x_i, \sigma_i)}{\sigma_i}.$$

Its simplicity makes it extremely fast, though it trades off by potentially underestimating complex dynamics in long trajectories [10].

- **k-LMS:** Adams–Bashforth uses k past slopes:

$$x_{i+1} = x_i + h_i \sum_{j=0}^{k-1} b_j \, f_{i-j}, \quad f_{i-j} = \frac{x_{i-j} - \hat{x}_0(x_{i-j}, \sigma_{i-j})}{\sigma_{i-j}}.$$

This higher-order multistep method balances additional memory needs against reduced step counts, but may require careful initialization and coefficient tuning for stability [11].

- **RK:** Runge–Kutta builds multiple intermediate slopes:

$$k_1 = f(x_i, \sigma_i),$$

$$k_2 = f\big(x_i + a_{21} h_i k_1, \; \sigma_i + c_2 h_i\big),$$

$$\vdots$$

$$k_k = f\Big(x_i + \sum_{j<k} a_{kj} h_i k_j, \; \sigma_i + c_k h_i\Big),$$

then

$$x_{i+1} = x_i + h_i \sum_{j=1}^{k} b_j \, k_j.$$

Classical RK4 remains a popular choice for its robust error control, although its multiple function calls can be costly [12].

- **DPM-Solver:** Exponential integrator that blends predictor and corrector: Predictor:

$$x^E_{i+1} = \sqrt{\alpha_{i+1}} \, \hat{x}_{0,i} + \sqrt{1 - \alpha_{i+1}} \, \epsilon_i,$$

Corrector:

$$x_{i+1} = \sqrt{\alpha_{i+1}} [\eta \, \hat{x}_{0,i} + (1 - \eta) \, \hat{x}_{0,i+1}],$$

where each stage draws on the denoiser's estimate to maintain stability with just two evaluations per step. Extensions to third or higher order use additional midpoint stages for reduced bias [14][15].

E. Network Architecture

Both actor and critic use MLP backbones. The actor is implemented as a denoiser network in the diffusion policy, taking noisy actions as additional input:

noitemsep

- **Actor (Diffusion Denoiser)** noitemsep
  - *Inputs:* a_t (noisy action), state vector s, diffusion timestep t (via 2-layer SinusoidalPosEmbed)
  - *Architecture:* concatenate $[a_t; s; \mathrm{Embed}(t)] \rightarrow$ 4-layer MLP
  - *Hidden units:* 256 per layer (512 for D4RL manipulation)
  - *Activation:* Mish
  - *Output:* predicted noise $\epsilon_\theta(a_t, s, t)$
- **Critic (Twin Q-Networks)** noitemsep
  - *Inputs:* state s, action a
  - *Architecture:* concatenate $[s; a] \rightarrow$ 4-layer MLP
  - *Hidden units:* 256 per layer (512 for D4RL manipulation)
  - *Activation:* Mish
  - *Output:* scalar $Q_\phi(s, a)$

F. Expert Pretraining and Fine-Tuning

In environments with challenging dynamics or high-dimensional control, the diffusion policy's denoiser and value networks may struggle to learn entirely from scratch. A supervised warm-start on expert demonstrations accelerates initial learning and stabilizes subsequent RL updates by embedding useful behavior priors into the actor.

noitemsep

1) **Expert dataset collection.** Gather a set of N_{demo} state–action pairs $\mathcal{D}_{\mathrm{expert}} = \{(s_i, a_i)\}$ from an oracle, scripted policy, or human demonstrations.
2) **Supervised denoising pretraining.** For K steps, sample $(s, a) \sim \mathcal{D}_{\mathrm{expert}}$, timestep t, and noise $\epsilon \sim \mathcal{N}(0, I)$. Construct the noisy action:

$$a_t = \sqrt{\bar{\alpha}_t} \, a + \sqrt{1 - \bar{\alpha}_t} \, \epsilon,$$

and update the actor to minimize:

$$\mathcal{L}_{\mathrm{pretrain}} = \big\| \epsilon - \epsilon_\theta(a_t, s, t) \big\|^2.$$

3) **Transition to RL fine-tuning.** After pretraining, resume the standard off-policy RL loop (actor and critic updates) using the DIPO or QVPO objectives. The replay buffer is seeded with both expert and random transitions to ensure continued exploration diversity.
4) **Trade-offs and limited impact.** While supervised pretraining can provide initial behavior priors, its impact on final performance was modest in practice, yielding only slight improvements when $K \leq 50{,}000$ and in some cases constraining exploration beyond the demonstration distribution.

IV. Experiments

A. Experimental Setup

The following environments were used to evaluate the diffusion-based RL methods:

a) OpenAI Gym MuJoCo Tasks: MuJoCo—short for Multi-Joint dynamics with Contact—is a high-fidelity physics engine designed for fast, accurate simulation of articulated bodies and contact dynamics [17]. OpenAI Gym exposes a suite of MuJoCo tasks (e.g. HalfCheetah-v2, Ant-v2, Hopper-v2) with continuous state and action spaces, serving as standard benchmarks for continuous-control algorithms [16].

b) D4RL Adroit Manipulation Tasks: D4RL (Datasets for Deep Data-Driven Reinforcement Learning) is an open-source benchmark suite providing environments and expert datasets tailored to offline and online RL research [18]. The Adroit subset simulates a 24-DoF Shadow Hand mounted on a 6-DoF arm performing dexterous manipulation. noitemsep

- *Pen Spin*: a 24-dimensional action space of normalized joint angles for wrist and finger articulations, defined on $\mathrm{Box}(-1, 1)$ [19].
- *Object Relocate*: a 30-dimensional action space controlling arm and hand joints, also on $\mathrm{Box}(-1, 1)$ [19].

Policies were trained online using the DIPO and QVPO objectives described in Section 3, with performance assessed via average episode return for locomotion tasks and success rates for manipulation tasks.

B. Fine-Tuning on D4RL Relocate

Two series of pretraining fine-tuning runs were conducted on the D4RL "Object Relocate" task, using DIPO and QVPO policies warm-started with $K = 10\,000$ and $K = 50\,000$ supervised denoising updates. In all cases the replay buffer contained both expert and random transitions, and training proceeded for 400 episodes of online RL. The fine-tuning of learning rates and soft update steps led to very minimal changes.

a) Reward performance: Figure 1 plots episodic return with $K = 10\,000$. Both DIPO (red) and QVPO (blue) fluctuate around –20 to –30 with no clear upward trend, indicating that this amount of pretraining was insufficient to bootstrap reliable object relocation.

Figure 2 shows returns with $K = 50\,000$. DIPO (green) exhibits a modest upward bias after episode 300, rising to around –15, while QVPO (orange) remains centered near –25. Even with heavier warm-start, neither method converges to positive returns, and performance remains highly variable.

b) Actor loss: Figure 3 compares the MSE denoiser loss for DIPO under three pretraining budgets: 0 (beige), 10k (red), and 50k (green). As K increases, the actor loss decreases sharply—50k pretraining yields the lowest noise-prediction error—confirming that supervised updates embed clearer expert priors into the policy denoiser.

c) Critic loss and stability: In contrast, the critic's TD-error (not shown) actually grows with larger K, suggesting that value estimates become less accurate when the policy is overly constrained by demonstrations. Additionally, both actor and critic losses under DIPO exhibit occasional large spikes and erratic fluctuations, reflecting unpredictable training dynamics introduced by gradient-guided sampling. QVPO losses show smoother curves but remain high enough to deny effective fine-tuning.

C. Fine-Tuning on D4RL Pen Spin

A series of pen-spin fine-tuning runs were conducted using default hyperparameters—batch size 256 and $n_{\text{timesteps}} = 40$, after confirming that varying batch sizes in $\{128,256,512\}$ and diffusion steps in $\{20,40,80\}$ had no significant impact on learning dynamics.

Fig. 1. Episodic returns on Relocate with 10k pretraining (DIPO: red; QVPO: blue).

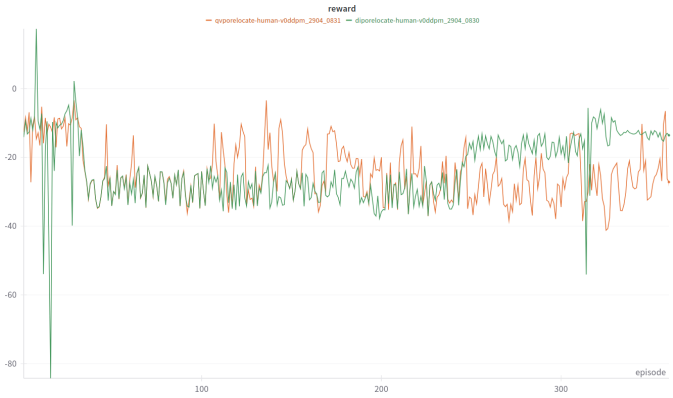

Fig. 2. Episodic returns on Relocate with 50k pretraining (DIPO: green; QVPO: orange).

a) Reward performance: Figure 4 shows episodic returns for DIPO (green) and QVPO (magenta) after $K = 10\,000$ supervised denoising updates. The reward is sparse—only achieved when the pen rotates through a full cycle—so the curves exhibit a characteristic sawtooth pattern, with sharp spikes followed by drops to zero. These occasional high-reward episodes suggest that the policy sometimes stumbles on a correct spin, indicating the task may be easier to bootstrap than object relocation, and could converge given more fine-tuning budget.

b) Actor and critic losses: The actor MSE loss under DIPO on the pen task follows a similar downward trend to the Relocate runs, confirming that the denoiser continues to improve noise-prediction accuracy. In contrast, QVPO's actor loss remains noisy and fails to settle, mirroring its unstable reward behavior. The critic TD-error for both methods fluctuates around a high baseline, with QVPO's critic loss roughly 10–20% higher on average, indicating value estimates are particularly unreliable under Q-weighting.

D. Sampler Comparison on Half-Cheetah

a) SDE-Based Samplers: Figure 5 shows the reward trajectories for three stochastic integrators. DDPM converges steadily to around 5 000–5 500, but exhibits moderate variance. Heun's method reaches similar returns with the

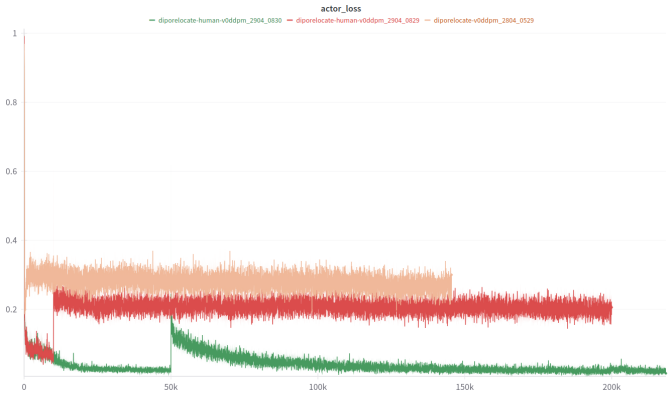

Fig. 3. Actor denoiser loss under 0 (beige), 10k (red), and 50k (green) pretraining steps.

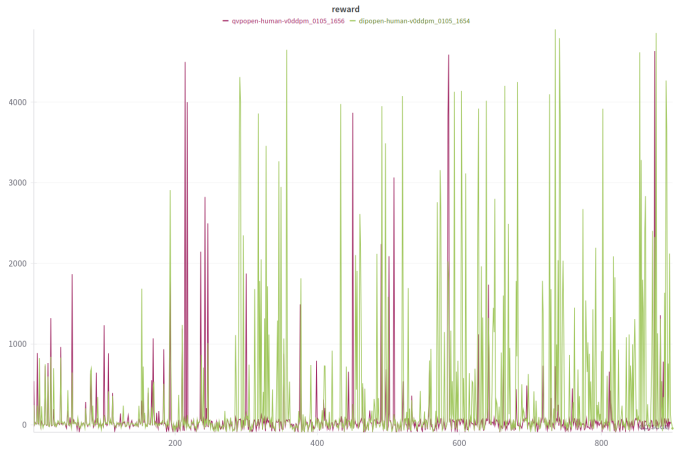

Fig. 4. Episodic returns on the Pen Spin task with 10k pretraining (DIPO: green; QVPO: magenta).

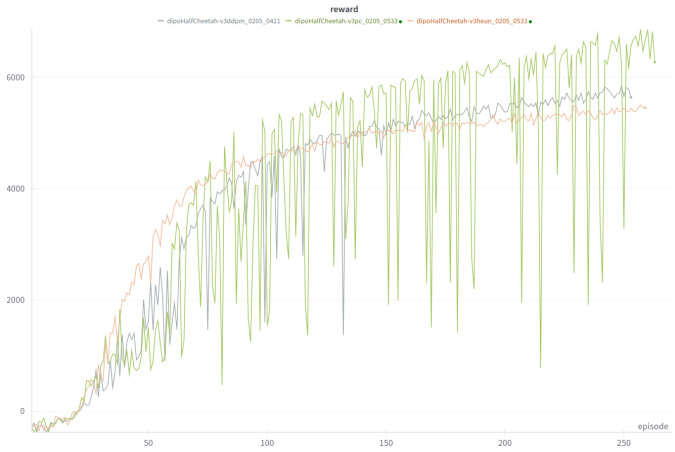

Fig. 5. Half-Cheetah returns for SDE-based samplers (DDPM: grey; PC: green; Heun: orange).

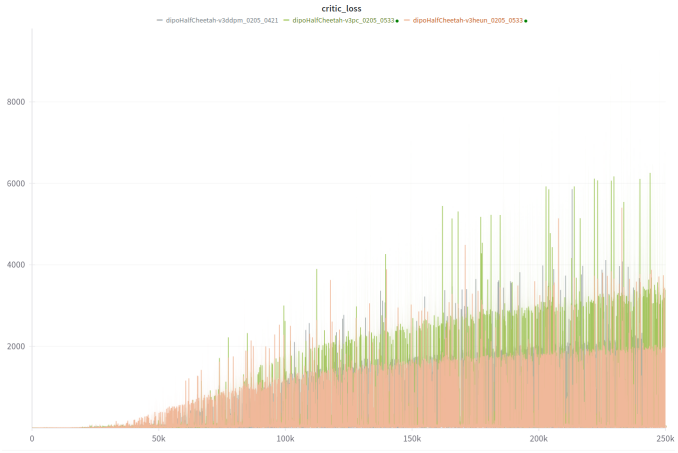

Fig. 6. Critic TD-error for SDE-based samplers (same color coding). Heun maintains the lowest error, while PC exhibits the largest fluctuations.

smoothest learning curve and lowest variance, indicating more accurate drift integration. The Predictor–Corrector (PC) sampler attains comparable mean returns but with frequent large spikes and collapses, reflecting its highest critic loss amplitudes (Fig. 6) and reduced value-estimate stability.

b) ODE-Based Samplers: Figure 7 compares four deterministic integrators. The k-step Linear Multistep (LMS, 4th order) and DPM-Solver (3rd order) rapidly climb to high returns with minimal variance, demonstrating the benefit of higher-order integration. DDIM (1st order) converges more slowly and with larger oscillations, consistent with its highest critic loss amplitude (Fig. 8). RK4 (4th order) delivers intermediate performance: more stable than DDIM but less smooth than LMS or DPM-Solver.

c) Overall Comparison: Across all seven samplers, the k-step Linear Multistep integrator demonstrates the best overall performance, combining rapid convergence, high average returns (above 6 500), and consistently low critic error. DPM-Solver follows closely, offering a favorable trade-off between accuracy and compute. In contrast, the Predictor–Corrector sampler exhibits the greatest variance and critic instability, and DDIM shows the slowest convergence and highest critic loss among the ODE methods. RK4 and DDPM occupy mid-range positions in both return and stability. A more definitive ordering would require longer training and additional random seeds to reduce residual noise.

V. CONCLUSIONS

VI. CONCLUSION

This work has introduced the use of alternative samplers for diffusion-based policies in the DIPO and QVPO frameworks and evaluated their effects both on standard locomotion benchmarks and on more challenging manipulation tasks. Among stochastic (SDE) samplers, Heun's method showed the most stable learning dynamics, while the Predictor–Corrector sampler suffered from high variance. For deterministic (ODE) samplers, higher-order integrators—particularly the k-step Linear Multistep and the DPM-Solver—delivered the best trade-off between convergence speed, final return, and critic stability, whereas DDIM lagged behind. These preliminary results suggest that sampler choice can substantially influence both variance reduction and sample efficiency in diffusion-policy training, but

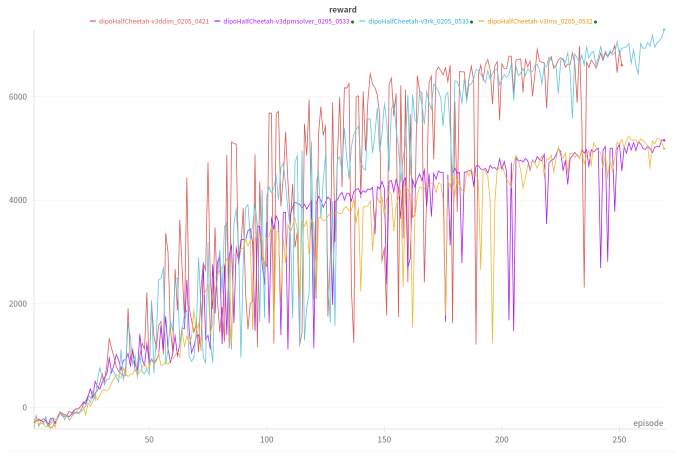

Fig. 7. Half-Cheetah returns for ODE-based samplers (DDIM: red; DPM-Solver: purple; RK4: teal; LMS: gold).

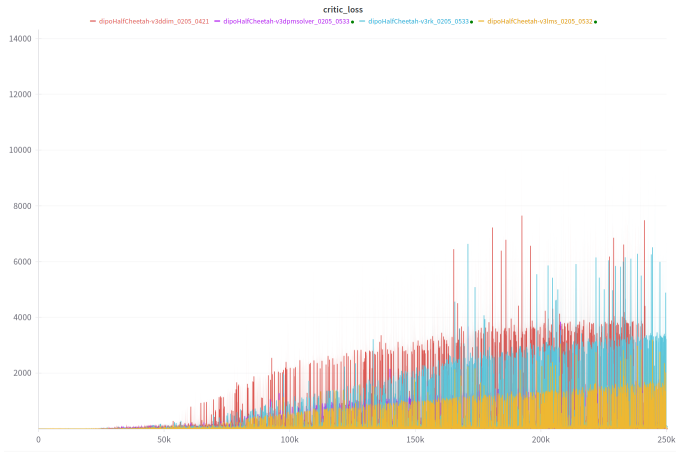

Fig. 8. Critic TD-error for ODE-based samplers (same color coding). LMS and DPM-Solver exhibit the lowest errors, while DDIM shows the highest.

more extensive experiments (longer training, additional random seeds, and varying environment difficulties) are needed to establish definitive rankings.

In the D4RL Adroit manipulation domains, fine-tuning with up to 50000 supervised pretraining steps yielded only modest improvements, indicating that complex, high-dimensional control may require more aggressive exploration strategies, larger or more specialized network architectures, and longer RL training budgets. Conversely, the pen-spin task exhibited promising sawtooth reward patterns and decreasing actor loss, suggesting that certain manipulation tasks can be effectively bootstrapped with light fine-tuning.

Overall, this analysis highlights both the potential and the current limitations of diffusion-based RL methods: sampler design is a powerful lever for improving stability and efficiency, but fully unlocking diffusion policies in complex environments will likely demand deeper investigation into exploration techniques, network scaling, and task-specific adaptations.

REFERENCES

[1] Yang, L., Huang, Z., Lei, F., Zhong, Y., Yang, Y., Fang, C., Wen, S., Zhou, B., & Lin, Z. (2023). *Policy Representation via Diffusion Probability Model for Reinforcement Learning*. arXiv:2305.13122.

[2] Ding, S., Hu, K., Zhang, Z., Ren, K., Zhang, W., Yu, J., Wang, J., & Shi, Y. (2024). *Diffusion-based Reinforcement Learning via Q-weighted Variational Policy Optimization*. NeurIPS 2024.

[3] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction* (2nd ed.). MIT Press, Cambridge, MA, 2018.

[4] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, 2014.

[5] Jonathan Ho and Stefano Ermon. Generative Adversarial Imitation Learning. In *Advances in Neural Information Processing Systems 29 (NeurIPS)*, pages 4565–4573, 2016.

[6] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision Transformer: Reinforcement Learning via Sequence Modeling. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, 2021.

[7] Ho, J., Jain, A. N., & Abbeel, P. (2020). *Denoising Diffusion Probabilistic Models*. NeurIPS.

[8] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the Design Space of Diffusion-Based Generative Models. In *Advances in Neural Information Processing Systems 35 (NeurIPS)*, 2022. :contentReference[oaicite:0]index=0

[9] 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 (ICLR)*, 2021. :contentReference[oaicite:1]index=1

[10] Song, J., Meng, C., & Ermon, S. (2021). *Denoising Diffusion Implicit Models*. ICLR.

[11] Liu, L., Ren, Y., Lin, Z., & Zhao, Z. (2022). *Pseudo Numerical Methods for Diffusion Models on Manifolds*. ICLR.

[12] Tim Salimans and Jonathan Ho. Progressive Distillation for Fast Sampling of Diffusion Models. In *International Conference on Learning Representations (ICLR)*, 2022. :contentReference[oaicite:0]index=0

[13] Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2021). *Score-Based Generative Modeling through Stochastic Differential Equations*. ICLR.

[14] Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., & Deng, J. (2022). *DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps*. NeurIPS.

[15] Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., & Deng, J. (2022). *DPM-Solver++: Fast Solver for Guided Sampling of Diffusion Probabilistic Models*. arXiv:2211.13458.

[16] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym. *arXiv preprint arXiv:1606.01540*, 2016.

[17] Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A Physics Engine for Model-Based Control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[18] 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.

[19] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations. In *Conference on Robot Learning (CoRL)*, 2018.