D2 Actor Critic: Diffusion Actor Meets Distributional Critic

Anonymous authors
Paper under double-blind review

Abstract

We develop a new model-free reinforcement learning (RL) algorithm: **D2AC**. Motivated by recent advances in iterative function approximation, we make two adjustments to the typical actor-critic RL pipeline. First, we learn distributional critics with a novel fusion of distributional RL and clipped double Q-learning. Second, we use a diffusion model to parameterize the policy and derive an efficient method for aligning the diffusion process with policy improvement. These changes are highly effective, resulting in highly performant model-free policies on a benchmark of eighteen hard RL tasks, including Humanoid, Dog, and Shadow Hand domains, spanning both dense-reward and goal-conditioned RL scenarios. Beyond standard benchmarks, we also evaluate a biologically motivated predator-prey task to examine the behavioral robustness and generalization capacity of our approach.

1 Introduction

Actor-Critic methods have been at the center of many recent advances in Reinforcement Learning. In continuous control and robotics, Soft Actor-Critic (Haarnoja et al., 2018) outperformed all prior model-based methods (Wang et al., 2019). On the ALE environment, Atari agents such as Rainbow (Hessel et al., 2018), BBF (Schwarzer et al., 2023) and ULTHO (Yuan et al., 2025) have continued to push forward state of the art reinforcement learning performance.

Each instantiation of actor-critic methods crucially relies on two components: an actor and a critic. The critic tells us how valuable it is to be in a given state, and the actor tells us what we should do when we find ourselves in that state. Generally, an actor-critic algorithm will only be as strong as the weakest link, a powerful critic is wasted on an actor that can't optimize for the correct actions, and vice-versa. This naturally leads us to ask the question of how we can use modern advancements in machine learning and function estimation to learn a critic and an actor that are a perfect match, equally powerful and capable of complementing one another.

When estimating the critic, classical RL favored point-estimates for the value function, often resulting in unstable or overly-opinated estimation, causing instability. Distributional critics (Bellemare et al., 2017) mitigate this problem by modeling distributional returns, making them a good deal more powerful than classical critics. In this paper, we explore the use and stability of distributional critics, and find that applying the clipped double Q-learning technique (Fujimoto et al., 2018) to distributional Q-function estimation can make the process more stable and reduce noise, improving the overall health of our critic.

Meanwhile, actor-learning is known to be a devilishly hard problem. The reasons for this are many and well-studied, but a major reoccuring issue is that action-estimates are simply very noisy. It's hard to tell in a single pass how we should generate an auto-correlated vector of actions in a complex environment. On this front, we consider the strategy of first producing a noisy estimate of what actions an agent should take, and then iteratively de-biasing this estimate, which gives the agent a powerful mechanism to propose and refine its actions.

More concretely, to get better action estimates we leverage denoising diffusion model for the policy network under the formulation of EDM (Karras et al., 2022), and derives a novel and simplified policy improvement

objective to supervise each step of the denoising process so that action proposals can be iteratively refined at inference. This derivation takes inspiration from the monotonic policy improvement theory (Schulman et al., 2015) and applies additional simplifications to the policy improvement bound based on properties of diffusion, yielding a conceptually simple yet fast and performant diffusion algorithm for policy improvement.

In this paper, we show how to marry a distributional critic to a diffusion actor, leading to the development of D2AC, a new actor-critic algorithm. D2AC achieves excellent performance on a variety of hard robotics environments, including locomotion tasks such as Humanoid and Dog domains in DeepMind Control Suite (Tassa et al., 2018), and manipulation tasks such as Pick-and-Place and Shadow-Hand Manipulate in multi-goal RL environments with sparse rewards (Plappert et al., 2018). In addition, we evaluate D2AC on a biology-inspired predator–prey benchmark(Lai et al., 2024), where it demonstrates higher exploration coverage, richer path diversity, and superior zero-shot transfer compared to SAC and TD-MPC2. Importantly, D2AC consistently performs well across difficult domains where prior model-free RL methods struggle or completely fail. Moreover, it significantly closes the performance gap between model-free RL and SOTA model-based methods such as TD-MPC2 (Hansen et al., 2023), despite using an order-of-magnitude less compute compared to TD-MPC2 and being considerably faster to run. Our results show that D2AC can serve as a strong base for policy optimization in continuous control. Videos of our policy in action can be found here [1].

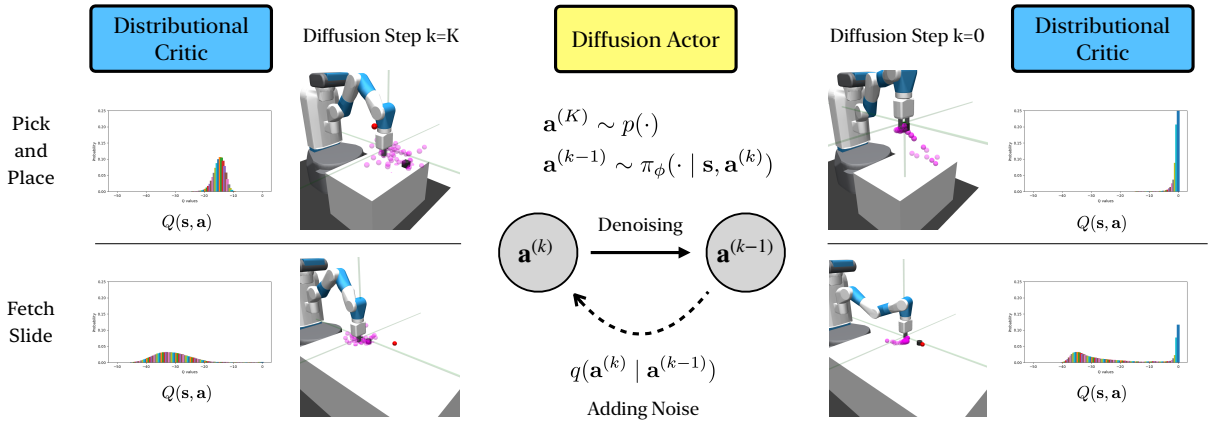

Figure 1: **D2 Actor Critic** uses a distributional Q-function to estimate a distribution over possible returns. A diffusion actor uses this Q-function to help align the denoising process with policy improvement. Above are visualizations on the Pick-and-Place and Fetch Slide environments.

2 Related Works

Model-free RL has two families of approaches: policy gradient methods and value-based methods. Policy gradient methods typically require on-policy data and trust-region regularization, as showcased in TRPO (Schulman et al., 2015), PPO (Schulman et al., 2017b), and IMPALA (Espeholt et al., 2018). Value-based methods, on the other hand, enable off-policy training, which can greatly improve sample-efficiency. Since DQN (Mnih et al., 2015), many improvements have been proposed including double Q-learning (Van Hasselt et al., 2016), clipped double Q-learning (Fujimoto et al., 2018), distributional RL (Bellemare et al., 2017; Dabney et al., 2018), and soft Q-learning (Haarnoja et al., 2018). Recent efforts further integrate uncertainty-aware critic learning for better exploration, such as Langevin Soft Actor-Critic (Ishfaq et al., 2025), which combines critic uncertainty with Langevin sampling for efficient policy updates. D2AC builds on this progress by unifying distributional critics with clipped double Q-learning and enabling iterative actor updates via a denoising process guided by value distributions.

Model Prediction Control (MPC) requires a learned dynamics model to generate virtual rollouts and plan its action accordingly. Popular methods include Random Shooting (RS) (Rao, 2009), Cross-Entropy

[1] `https://d2ac-actor-critic.github.io/`

Method (CEM) (De Boer et al., 2005), and Model Predictive Path Integral Control (MPPI) (Williams et al., 2015a). MPC is also closely related to model-based RL framework Dyna (Sutton, 1991). Traditionally, the pain point of MPC is the quality of the learnt dynamics: model-based methods had lower asymptotic performance on hard domains (Wang et al., 2019), where learning good world models requires more expressive generative models (Janner et al., 2021; Zhang et al., 2023). MuZero (Schrittwieser et al., 2020) managed to circumvent this issue by only modeling future rewards and values without the states. Sampled MuZero (Hubert et al., 2021), TD-MPC (Hansen et al., 2022) all combine a model-free agent, MuZero dynamics, and an off-the-shelf planning module to achieve impressive results on hard tasks in continuous control. Our work is inspired by MPC, but only tackles model-free RL. D2AC is orthogonal to model-based planning, and can be freely combined with any dynamics learning and MPC method.

Diffusion Policy for RL has attracted interest following the success of denoising diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020) and its generalized formulation, denoising score matching (DSM) (Song et al., 2020b; 2023; Chewi et al.). The most straightforward application of such models is conditional generative modeling of actions, or conditional behavior cloning (BC), using a denoising policy (Janner et al., 2022; Chi et al., 2023; Römer et al., 2024). However, making denoising policies compatible with the online RL has been a challenge, because policy improvement necessarily means that the target data distribution is non-stationary. Existing works have studied how to apply diffusion policies to offline RL settings (Hansen-Estruch et al., 2023; Gao et al., 2025) and online RL settings (Chen et al., 2023; Ma et al., 2025; Dong et al., 2025). Our work uses the monotonic improvement theory from policy optimization literature (Schulman et al., 2015) to derive a new method of training diffusion policies in online RL setting, which significantly improves upon prior art (Yang et al., 2023).

3 Background

3.1 Reinforcement Learning

A Markov Decision Process (MDP) is defined by the state space $\mathcal{S}$, action space $\mathcal{A}$, the initial state distribution $\rho_0(\mathbf{s})$, the transition probability $p(\mathbf{s}' \mid \mathbf{s}, \mathbf{a})$, and the reward function $r(\mathbf{s}, \mathbf{a})$.

The policy $\pi(\mathbf{a} \mid \mathbf{s})$ generates a trajectory τ with horizon T within the environment:

$$\tau = \{\mathbf{s}_0, \mathbf{a}_0, \cdots, \mathbf{s}_T\} \tag{1}$$

where $\mathbf{s}_0 \sim \rho_0(\cdot)$, $\mathbf{a}_t \sim \pi(\cdot \mid \mathbf{s}_t)$, and $\mathbf{s}_{t+1} \sim p(\cdot \mid \mathbf{s}_t, \mathbf{a}_t)$ for $t \in \{0, \cdots, T-1\}$.

We denote the trajectory distribution starting from state s under policy π as $p_\pi(\tau \mid \mathbf{s})$. With a discount factor $\gamma \in (0, 1)$, the goal is for the policy π to maximize the discounted cumulative rewards:

$$\mathcal{J}(\pi) = \mathbb{E}_{\rho_0(\mathbf{s}_0) p_\pi(\tau|\mathbf{s}_0)} \left[\sum_{t=0}^{T-1} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t) \right] \tag{2}$$

Value-based RL methods optimize this objective by estimating a Q function for the current policy π:

$$Q(\mathbf{s}_t, \mathbf{a}_t) = \mathbb{E}_{p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) p_\pi(\tau|\mathbf{s}_{t+1})} \left[\sum_{\Delta=0}^{T-t} \gamma^\Delta r(\mathbf{s}_{t+\Delta}, \mathbf{a}_{t+\Delta}) \right] \tag{3}$$

and then training the policy π to maximize Q.

3.2 Denoising Diffusion Models

Denoising Diffusion Models are generative models that estimate the gradients of data distribution (Vincent, 2011; Song & Ermon, 2019; Song et al., 2020a). Diffusion models train a denoiser D_ϕ to reconstruct data $\mathbf{x} \sim p_{\text{data}}$ from noised data $\mathbf{x}^{(k)} \sim \mathcal{N}(\mathbf{x}, \sigma_k^2 \mathbf{I})$ under various noise levels $\sigma_{\max} = \sigma_K > \cdots > \sigma_0 = \sigma_{\min}$.

Let the distribution of noised data be defined as:

$$p(\mathbf{x}^{(k)}) = \int p_{\text{data}}(\mathbf{x}) \mathcal{N}(\mathbf{x}^{(k)} | \mathbf{x}, \sigma_k^2 \mathbf{I}) \mathrm{d}\mathbf{x} \tag{4}$$

The min $\sigma_{\min}$ and max $\sigma_{\max}$ of noise levels are selected such that $p(\mathbf{x}^{(0)}) \approx p_{\text{data}}(\mathbf{x})$ and $p(\mathbf{x}^{(K)}) \approx \mathcal{N}(\mathbf{0}, \sigma_{\max}^2 \mathbf{I})$.

The diffusion loss is typically:

$$\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),\, k\sim\text{Unif}(1\cdots K)}[\lambda(k)\|D_\phi(\mathbf{x}+\boldsymbol{\epsilon}\sigma_k;\sigma_k)-\mathbf{x}\|^2] \tag{5}$$

where $\lambda(k)$ is the loss weighting based on noise level.

Then the score for arbitrary input $\mathbf{x}$ is:

$$\nabla_{\mathbf{x}} \log p(\mathbf{x};\sigma) = \frac{D_\phi(\mathbf{x};\sigma)-\mathbf{x}}{\sigma^2} \tag{6}$$

which can be used to sample data by starting from noise and solving either an SDE or ODE (Song et al., 2020b).

We follow EDM (Karras et al., 2022) to define the denoiser as:

$$D_\phi(\mathbf{x},\sigma) = c_{\text{skip}}(\sigma)\mathbf{x} + c_{\text{out}}(\sigma)F_\phi(c_{\text{in}}(\sigma)\mathbf{x}, c_{\text{noise}}(\sigma)) \tag{7}$$

3.3 Clipped Double Distributional Q-Learning

While distributional reinforcement learning (RL) provides richer return modeling by capturing full return distributions instead of scalar estimates (Bellemare et al., 2017), it inherits a well-known flaw of traditional Q-learning: overestimation bias (Van Hasselt, 2010; Van Hasselt et al., 2016). In distributional RL, this bias manifests not just in the expected value, but across entire regions of the predicted return distribution, leading to pathological optimism.

To address this, we apply a clipped double Q-learning approach (Fujimoto et al., 2018) within the distributional setting. Though this combination is conceptually simple, it plays a crucial role in stabilizing training and improving empirical performance.

Categorical Distributional Q-Functions: We represent return distributions using the categorical formulation introduced in (Bellemare et al., 2017). Specifically, each Q-function $\mathbf{q}_\theta(\mathbf{s},\mathbf{a})$ outputs a softmax distribution over a fixed support $\mathbf{z}_q = [V_{\min},\ldots,V_{\max}]$ consisting of $N+1$ discrete bins. The expected Q-value is computed as:

$$Q_\theta(\mathbf{s},\mathbf{a}) = \mathbf{q}_\theta(\mathbf{s},\mathbf{a})^\top \mathbf{z}_q$$

Any continuous value within the support range can be represented via a linear combination of two adjacent bins using a *two-hot* encoding function $h_{\mathbf{z}_q}(\cdot)$.

Categorical Projection: To perform bootstrapped learning, we shift the support according to the observed reward and discount factor: $\mathbf{z}_p = r + \gamma \mathbf{z}_q$. The predicted distribution $\mathbf{p} = \mathbf{q}_{\overline{\theta}}(\mathbf{s}',\mathbf{a}')$ under this shifted support must be projected back to the original support $\mathbf{z}_q$ to compute a training target. This is done via:

$$\boldsymbol{\Phi}_{\text{dist}}(\mathbf{z}_p,\mathbf{p},\mathbf{z}_q)[k] = \sum_{j=0}^{N} h_{\mathbf{z}_q}(\mathbf{z}_p[j])[k]\cdot\mathbf{p}[j]$$

This yields a valid probability distribution over the original support, which serves as a supervised signal.

Clipped Double Distributional RL: To reduce overestimation, we maintain two independent critics, $\mathbf{q}_{\theta_1}$ and $\mathbf{q}_{\theta_2}$, each producing a distribution over returns. For each training step, we compute expected Q-values:

$$Q_{\theta_i}(\mathbf{s},\mathbf{a}) = \mathbf{q}_{\theta_i}(\mathbf{s},\mathbf{a})^\top \mathbf{z}_q$$

We then select the entire distribution from the critic with the lower expected value:

$$\mathbf{q}_\theta^{\text{clip}}(\mathbf{s},\mathbf{a}) = \begin{cases} \mathbf{q}_{\theta_1}(\mathbf{s},\mathbf{a}) & \text{if } Q_{\theta_1} \le Q_{\theta_2} \\ \mathbf{q}_{\theta_2}(\mathbf{s},\mathbf{a}) & \text{otherwise} \end{cases}$$

This clipped distribution is projected and used to train both critics via cross-entropy loss. The full critic loss becomes:

$$L_{\text{critic}}(\theta) = -\boldsymbol{\ell}^\top \left(\log \mathbf{q}_{\theta_1}(\mathbf{s}, \mathbf{a}) + \log \mathbf{q}_{\theta_2}(\mathbf{s}, \mathbf{a})\right)$$

where $\boldsymbol{\ell} = \boldsymbol{\Phi}_{\text{dist}}(r + \gamma \mathbf{z}_q, \mathbf{q}^{\text{clip}}_{\bar{\theta}}(\mathbf{s}', \mathbf{a}'), \mathbf{z}_q)$ and $\mathbf{a}' \sim \pi_\phi(\cdot \mid \mathbf{s}')$.

4 Policy Improvement for Diffusion Actors

When the policy π in actor-critic algorithms is a diffusion model (Sohl-Dickstein et al., 2015; Ho et al., 2020), it starts from the noise distribution $\mathcal{N}(0, \sigma^2_{\max}\mathbf{I})$ and goes through K diffusion steps $\mathbf{a}^{(K)} \to \mathbf{a}^{(K-1)} \cdots \to \mathbf{a}^{(0)}$ using a sequence of noise levels $\sigma_{\max} = \sigma_K > \cdots > \sigma_1 > \sigma_0 = \sigma_{\min}$ to sample an action. If we assume a sampler based on the Euler-Maruyama method (Song et al., 2020b), then one step of the diffusion process $\mathbf{a}^{(k)} \to \mathbf{a}^{(k-1)}$ for policy π_ϕ can be written as:

$$\pi_\phi(\mathbf{a}^{(k-1)} \mid \mathbf{s}, \mathbf{a}^{(k)}) = \mathcal{N}(\mathbf{a}^{(k-1)} \mid \boldsymbol{\mu}_\phi(\mathbf{a}^{(k)}, k; \mathbf{s}), (\sigma_k^2 - \sigma_{k-1}^2)\mathbf{I})$$

Where $\boldsymbol{\mu}_\phi$ is a learned denoising process that takes actions away from the noise and towards the desired action distribution. It is parameterized as:

$$\begin{aligned}
\boldsymbol{\mu}_\phi(\mathbf{a}^{(k)}, k; \mathbf{s}) &= \mathbf{a}^{(k)} + (\sigma_k^2 - \sigma_{k-1}^2)\nabla_{\mathbf{a}} \log p(\mathbf{a}^{(k)} \mid \sigma_k, \mathbf{s}) \\
\nabla_{\mathbf{a}} \log p(\mathbf{a} \mid \sigma, \mathbf{s}) &= (D_\phi(\mathbf{a}, \sigma; \mathbf{s}) - \mathbf{a})/\sigma^2
\end{aligned}$$

where $D_\phi(\mathbf{a}, \sigma; \mathbf{s}) = c_{\text{skip}}(\sigma)\mathbf{a} + c_{\text{out}}(\sigma)F_\phi(\mathbf{s}, c_{\text{in}}(\sigma)\mathbf{a}, c_{\text{noise}}(\sigma))$. The functions $c_{\text{skip}}(\sigma)$, $c_{\text{out}}(\sigma)$, and $c_{\text{noise}}(\sigma)$ are flexible design choices (Karras et al., 2022). Thus, taking an action $\mathbf{a}^{(0)}$ from the denoising process $\mathbf{a}^{(k)} \to \cdots \mathbf{a}^{(0)}$ can be written as an integral over the denoising steps:

$$\pi_\phi(\mathbf{a} \mid \mathbf{s}) = \int p(\mathbf{a}^{(K)}) \prod_{k=1}^{K} \pi_\phi(\mathbf{a}^{(k-1)} \mid \mathbf{s}, \mathbf{a}^{(k)}) \mathrm{d}\mathbf{a}_{1:K} \tag{8}$$

We would like a way to guarantee the diffusion process actually provides us with a better action. Define the optimal policy to be $p^*(\mathbf{a} \mid \mathbf{s}) = (\exp Q(\mathbf{s}, \mathbf{a})/\alpha)/Z(\mathbf{s})$, where α is the temperature. The goal of entropy-regularized policy improvement is to minimize the following (Schulman et al., 2017a; Haarnoja et al., 2018; Black et al., 2023):

$$\arg\max_\phi - D_{KL}(\pi_\phi(\mathbf{a} \mid \mathbf{s}) \parallel p^*(\mathbf{a} \mid \mathbf{s})) = \mathbb{E}_{\mathbf{a} \sim \pi_\phi(\cdot \mid \mathbf{s})}[Q(\mathbf{s}, \mathbf{a})] + \alpha H(\pi_\theta(\mathbf{a} \mid \mathbf{s})) \tag{9}$$

Diffusion Policy Gradients: In the case of diffusion policies, our main challenge with optimizing for policy improvement is that accessing $\mathbf{a} \sim \pi_\phi$ requires K intermediate sampling steps. To optimize a lower bound on the policy gradient objective, the variational lower bound in (Kingma et al., 2021) can be applied with a positive transformation of the Q-values (Peters & Schaal, 2007; Abdolmaleki et al., 2018), with $\lambda_{\text{pg}}(k) = (1/\sigma_{k-1}^2 - 1/\sigma_k^2) \cdot (K/2)$:

$$\mathbb{E}_\pi \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}), k \sim \text{Unif}(1 \cdots K)}[\lambda_{\text{pg}}(k) \| \mathbf{a} - D_\phi(\mathbf{a} + \sigma_k \boldsymbol{\epsilon}, \sigma_k; \mathbf{s}) \|^2 \cdot \exp Q(\mathbf{s}, \mathbf{a})] \tag{10}$$

On the one hand, policy gradients tend to suffer from high variance and instability. On the other hand, value gradients (Heess et al., 2015) for diffusion require backpropagation through time (BPTT), which is known to be difficult (Pascanu et al., 2013). Those challenges have been previously noted in (Wallace et al., 2023; Black et al., 2023; Clark et al., 2023). Next, we will show that by re-examining the MDP view of diffusion process and making mild assumptions, we can circumvent those challenges. The main insight is that we can convert a classical policy improvement result from the literature into an easier-to-manage form by isolating the effect of one-step denoising.

Diffusion MDP: Towards a policy improvement result for a diffusion actor, let's first define a diffusion MDP as taking K steps of actions from $p(\mathbf{a}^{(K)})$ to $\pi(\mathbf{a}^{(k-1)} \mid \mathbf{s}, \mathbf{a}^{(k)})$, reaching the final state $\mathbf{a}^{(0)}$. After this

diffusion process (under discount factor γ), the policy receives a reward $Q(\mathbf{s}, \mathbf{a}^{(0)})/\gamma^K$ at the final timestep of diffusion (reward is 0 elsewhere). The objective of diffusion is to maximize:

$$\eta(\mathbf{s}, \pi) = \mathbb{E}_{\mathbf{a}^{(K)} \cdots \mathbf{a}^{(0)} \sim \pi}[Q(\mathbf{s}, \mathbf{a}^{(0)})] \tag{11}$$

Where we have made the dependence on the intermediate diffusion steps $\mathbf{a}^{(K)} \cdots \mathbf{a}^{(1)}$ explicit. In the diffusion MDP, the Q-function, value function, and advantage function are given by:

$$\begin{aligned}
V_\pi^{\text{diffusion}}((\mathbf{s}, \mathbf{a}^{(k)})) &\triangleq \mathbb{E}_{\mathbf{a}^{(k-1)} \cdots \mathbf{a}^{(0)} \sim \pi}[(\gamma^{k-1}/\gamma^K) Q(\mathbf{s}, \mathbf{a}^{(0)})] \\
Q_\pi^{\text{diffusion}}((\mathbf{s}, \mathbf{a}^{(k+1)}), \mathbf{a}^{(k)})) &\triangleq \mathbb{E}_{\mathbf{a}^{(k-1)} \sim \pi}[\gamma V_\pi^{\text{diffusion}}((\mathbf{s}, \mathbf{a}^{(k-1)}))] \\
A_\pi^{\text{diffusion}}((\mathbf{s}, \mathbf{a}^{(k+1)}), \mathbf{a}^{(k)})) &\triangleq Q_\pi^{\text{diffusion}}((\mathbf{s}, \mathbf{a}^{(k+1)}), \mathbf{a}^{(k)})) - V_\pi^{\text{diffusion}}((\mathbf{s}, \mathbf{a}^{(k+1)}))
\end{aligned}$$

Monotonic improvement theory, and in particular the policy improvement objective from Trust Region Policy Optimization (Schulman et al., 2015), can now be applied naively to the diffusion MDP framework. Doing so will give us gives us the following proximal objective for policy improvement:

$$\begin{aligned}
\eta(\mathbf{s}, \pi) &\geq J_{\pi_{\text{ref}}}(\mathbf{s}, \pi) - \frac{4\epsilon\gamma}{(1-\gamma)^2} \left\{ \max_{k, \mathbf{a}^{(k)}} D_{KL}(\pi_{\text{ref}}(\cdot \mid \mathbf{s}, \mathbf{a}^{(k)}), \pi(\cdot \mid \mathbf{s}, \mathbf{a}^{(k)})) \right\} \\
J_{\pi_{\text{ref}}}(\mathbf{s}, \pi) &\triangleq \eta(\mathbf{s}, \pi_{\text{ref}}) + \mathbb{E}_{\substack{k \sim \text{Unif}\{1,K\}, \mathbf{a}^{(k)} \sim \pi_{\text{ref}} \\ \mathbf{a}^{(k-1)} \sim \pi(\cdot \mid \mathbf{a}^{(k)}, \mathbf{s})}}[\gamma^k A_{\pi_{\text{ref}}}^{\text{diffusion}}((\mathbf{s}, \mathbf{a}^{(k)}), \mathbf{a}^{(k-1)})]
\end{aligned}$$

These equations say that we can lower bound our policy objective equation 11, by a loss that depends on the Diffusion MDP advantage of the reference policy $A_{\pi_{\text{ref}}}^{\text{diffusion}}$, and the KL between the reference policy π_{ref} and the policy we are learning π. Essentially, if π and π_{ref} are sufficiently close, then if π_{ref} makes an improvement, so does π. Since this is a lower bound on the target objective equation 11, we can move in the direction of improvement by taking the gradient of $J_{\pi_{\text{ref}}}(\mathbf{s}, \pi)$ in the lower bound

$$\nabla_\phi J_{\pi_{\text{ref}}}(\mathbf{s}, \pi_\phi) = \nabla_\phi \underbrace{\mathbb{E}_{k \sim \text{Unif}\{1,K\}, \mathbf{a}^{(k)} \sim \pi_{\text{ref}}} \left[\gamma^k \mathbb{E}_{\mathbf{a}^{(k-1)} \sim \pi_\phi}[Q_{\pi_{\text{ref}}}^{\text{diffusion}}((\mathbf{s}, \mathbf{a}^{(k)}), \mathbf{a}^{(k-1)})] \right]}_{J_{\pi_{\text{ref}}}^{\text{proximal}}(\mathbf{s}, \pi_\phi)} \tag{12}$$

Note that in practice this is still difficult, because we need to learn a separate Q function $Q^{\text{diffusion}}$.

Proposed simplification: Rather than learning $Q_{\pi_{\text{ref}}}^{\text{diffusion}}$ for diffusion MDP, we utilize the fact that the denoising function $\hat{\mathbf{a}} = D_\phi(\mathbf{a}^{(k)}, \sigma; \mathbf{s})$ directly aims to predict a strictly cleaner signal with each passing step (Song et al., 2020a). The diffusion literature has shown that (Ho et al., 2020; Song et al., 2020b) for a well-trained diffusion model, more sampling steps result in smaller truncation error in the SDE solver and higher quality of $\hat{\mathbf{a}}$. Since Q-value is analogous to unnormalized log probability (Levine, 2018), it is plausible to reason that, under mild assumptions, *more sampling steps should lead to higher Q-value*, giving us a lower bound on the Q-value based on a single denoising step. We formalize this intuition in the following proposition.

Proposition 1. *Define the data distribution under diffusion step $k \geq 0$ to be (with $\hat{\sigma} \geq \sigma_{min}$):*

$$p_k(\mathbf{a}|\mathbf{s}) = \int \pi(\mathbf{a}^{(0)}|\mathbf{s}) \mathcal{N}(\mathbf{a}|\mathbf{a}^{(0)}, \sigma_k^2 \mathbf{I}) \mathrm{d}\mathbf{a}^{(0)} \quad \hat{p}_k(\mathbf{a}|\mathbf{s}) = \int p_{k+1}(\mathbf{u}|\mathbf{s}) \mathcal{N}(\mathbf{a}|D(\mathbf{u}, \sigma_{k+1}; \mathbf{s}), \hat{\sigma}^2 \mathbf{I}) \mathrm{d}\mathbf{u}$$

Under the assumptions that (i) entropy of p_k and $\hat{p}_k$ strictly increases with k; (ii) the KL distance from p_k and $\hat{p}_k$ to optimal policy p^* strictly decreases with k; (iii) $\hat{\sigma}$ is sufficiently large such that $D_{KL}(\hat{p}_0 \parallel p^*) \geq D_{KL}(p_0 \parallel p^*)$. Then for any state $\mathbf{s}$ and diffusion step k, we have

$$\mathbb{E}_{p_0}[Q] \geq \mathbb{E}_{\hat{p}_k}[Q] \tag{13}$$

Which results in the following lower bound of equation equation 12 based on one-step generation:

$$\begin{aligned}
&\gamma^k \mathbb{E}_{\mathbf{a}^{(k-1)} \sim \pi_\phi}[Q_{\pi_{\text{ref}}}^{\text{diffusion}}((\mathbf{s}, \mathbf{a}^{(k)}), \mathbf{a}^{(k-1)})] \\
=&\gamma^{2k-K} \mathbb{E}_{\mathbf{a}^{(k-1)} \sim \pi_\phi} \underbrace{\mathbb{E}_{\mathbf{a}^{(k-2)} \cdots \mathbf{a}^{(0)} \sim \pi_{\text{ref}}}}_{\text{Additional } k-1 \text{ SDE steps}} [Q(\mathbf{s}, \mathbf{a}^{(0)})] \geq \gamma^{2k-K} \underbrace{\mathbb{E}_{\mathcal{N}(\hat{\mathbf{a}}|D_\phi(\mathbf{a}^{(k)}, \sigma_k; \mathbf{s}); \hat{\sigma}^2)}}_{\text{One step generation}} [Q(\mathbf{s}, \hat{\mathbf{a}})]
\end{aligned}$$

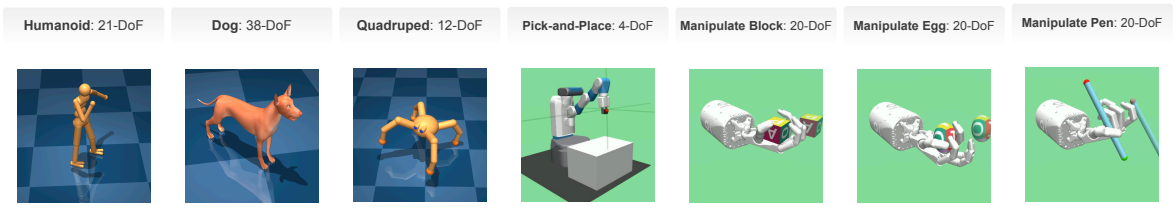

Figure 2: D2AC works out of the box across a wide range of environments, including locomotion and manipulation with sparse and dense rewards.

Intuitively, having additional refinement steps on action $\hat{\mathbf{a}}$ should result in higher Q-value on average.

Diffusion Value Gradients: This lower bound based on one-step generation is much easier to optimize, since it avoids the requirement for separate value functions $Q^{\text{diffusion}}_{\pi_{\text{ref}}}$ in diffusion MDP.

$$J^{\text{proximal}}_{\pi_{\text{ref}}}(\mathbf{s}, \pi_\phi) \geq \mathbb{E}_{k\sim\text{Unif}\{1,K\},\mathbf{a}^{(k)}\sim\pi_{\text{ref}},\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})}[(\gamma^{2k}/\gamma^K)\cdot Q(\mathbf{s}, D_\phi(\mathbf{a}^{(k)}, \sigma_k; \mathbf{s}) + \hat{\sigma}\boldsymbol{\epsilon})] \tag{14}$$

This simplified policy objective can be written in a form similar to the L_2 loss commonly used in diffusion models (Kingma et al., 2021). In essence, the loss takes a noised version of on-policy action, and learns how to iteratively improve upon this noised action via denoising guided by the critic Q. Each intermediate diffusion step is aligned towards policy improvement. Setting $\pi_{\text{ref}} = \pi_\phi$, the D2AC policy loss is:

$$\nabla_\phi L^{\text{simple}}_\pi(\phi) = \mathbb{E}_{\substack{\mathbf{a}\sim\pi_\phi(\cdot|\mathbf{s}) \\ \boldsymbol{\epsilon}\sim\mathcal{N}(0,I),k\sim\text{Unif}\{1,K\} \\ \mathcal{N}(\hat{\mathbf{a}}|D_\phi(\mathbf{a}+\sigma_k\boldsymbol{\epsilon},\sigma_k;\mathbf{s}),\hat{\sigma}^2)}} \nabla_\phi \left[\lambda(k)\left\|\hat{\mathbf{a}} + \nabla_{\hat{\mathbf{a}}}Q(\mathbf{s},\hat{\mathbf{a}}) - D_\phi(\mathbf{a}+\sigma_k\boldsymbol{\epsilon},\sigma_k;\mathbf{s})\right\|^2\right] \tag{15}$$

The function $\lambda(k)$ is the weighting according to the noise schedule. $\lambda(k) = \gamma^{2k}/\gamma^K$ in our derivation; in practice we find that simply setting $\lambda(k) = 1$ works well. Compared to diffusion policy gradients equation 17, our diffusion value gradients are designed to directly align the noisy predictions based on value gradients from Q without backprop-through-time, thus providing cleaner supervision signal.

5 Experiments

In these experiments, we evaluate the performance of D2AC across a range of domains to assess its generality, efficiency, and behavioral characteristics. Our benchmarks cover various task modalities such as locomotion, grasping, and manipulation, as well as both sparse and dense reward structures. In addition to standard control tasks, we introduce a biology-inspired predator–prey environment to examine how D2AC performs scenarios that demand strategic adaptability.

We also investigate how D2AC compares to strong model-based baselines in selected tasks, aiming to understand how effectively it bridges the gap between model-free and model-based approaches. Finally, we evaluate the algorithm's practical scalability by measuring wall-clock training time and analyzing how its performance varies with compute resources. We use rliable (Agarwal et al., 2021) library for rigorous statistical evaluation.

Evaluation environments. We evaluate D2AC across three primary domains: (i) the DeepMind Control Suite (Tassa et al., 2018), which consists mainly of locomotion tasks with dense rewards, (ii) the Multi-Goal RL environments from (Plappert et al., 2018), which include robotic manipulation tasks with sparse rewards, and (iii) a predator–prey environment inspired by biological survival dynamics (Lai et al., 2024), which emphasizes adaptive behavior in dynamic and high-stakes scenarios. A list of the environments we use are shown in Figure 2. Those environments include 21-DoF Humanoid, 38-DoF Dog, 12-DoF Quadruped, a 4-DoF Pick-and-Place task, and a series of 20-DoF Shadow Hand tasks aiming to manipulate Block, Egg, and Pen to target orientations.

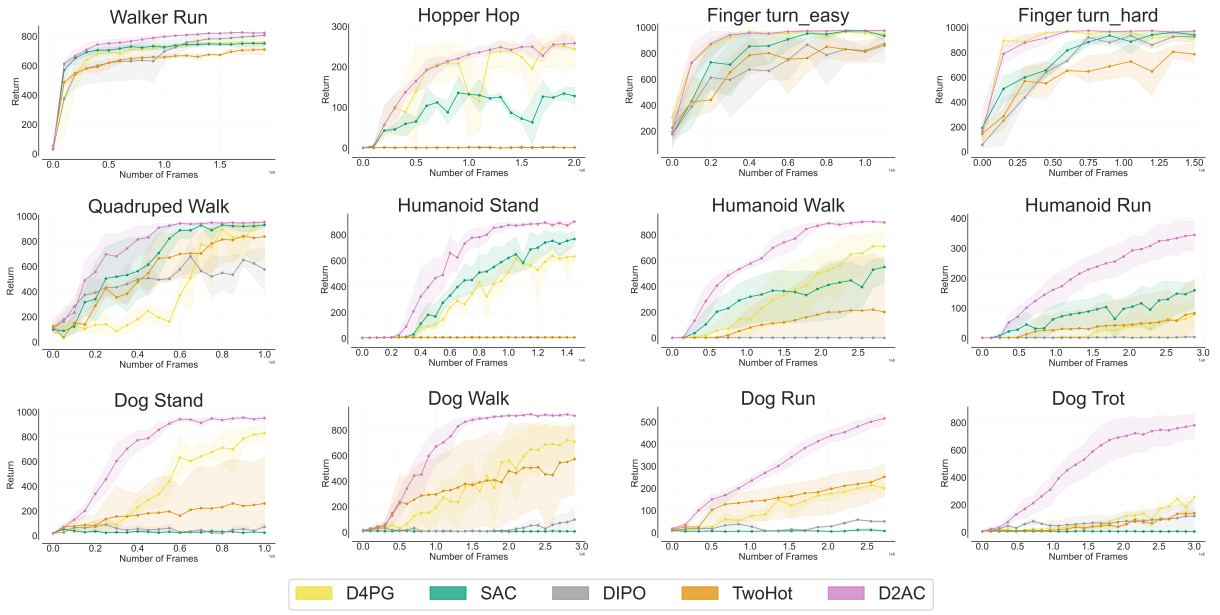

Figure 3: Experiments on **DeepMind Control Suite**. Results over 5 seeds. In model-free RL, D2AC achieves much better sample efficiency and asymptotic performance compared to all other baselines.

5.1 Benchmarking on dense-reward environments

In Figure 3, we showcase our benchmarking results on 12 tasks in DeepMind control suite. We compare our method to a variety of baselines, including: Soft Actor Critic (Haarnoja et al., 2018), D4PG (Barth-Maron et al., 2018), and the model-free component of TD-MPC2 (Hansen et al., 2023), which we denote as *Two-hot + CDQ*, because it essentially combines SAC, two-hot critic representation, and clipped double Q-learning. We also compare against DIPO (Yang et al., 2023), which is a model-free RL method based on diffusion policy. D2AC achieves higher asymptotic performance across all 12 environments, and generally learns faster.

When compared to TD-MPC2 (Hansen et al., 2023) trained with the same number of environment interactions, we see that D2AC is very competitive despite being model-free. In Figure 4, we see that D2AC achieves performance within five percent of TD-MPC2 on 8 out of 12 benchmark environments. On 4 of the environments, the performance of D2AC exceeds TD-MPC2. The two standout environments were Dog Walk and Humanoid Stand, where D2AC significantly outpaced TD-MPC2 performance. The relatively poor performance of *Two-hot + CDQ* baseline shows that the model-free component of TD-MPC2 is not in isolation a strength of the algorithm, and the strong performance of TD-MPC2 relies on its ability to pair this model free algorithm with its planning module. Our proposed model-free learning algorithm is orthogonal to any planning algorithm, and can be used as a drop-in replacement in any model-based method such as TD-MPC2.

D2AC Is highly performant in a wide variety of settings: D2AC achieves significantly better sample efficiency and asymptotic performance compared to all other model-free baselines we tested. This is true across multiple control modalities (grasping, locomotion, manipulation). We use 5 seeds for the results shown in Figure 3 and Figure 4; we also use the rliable (Agarwal et al., 2021) evaluation under Interquartile Mean (IQM) with interval estimates via stratified bootstrap confidence intervals.

Thanks to being model-free, **our method is considerably faster to run compared to TD-MPC2**: see Figure 9 for a clear comparison. In particular, D2AC is able to make significant process on the Dog Trot task within 24 hours, where TD-MPC2 has barely started increasing its episodic return; SAC completely fails on this task.

Overall, the strong performance of D2AC seems to suggest that model-based RL's recent success does not stem entirely from planning with rollouts. Rather, a major component of this success is model-based RL's ability to model a distribution of returns and iteratively refine action proposals.

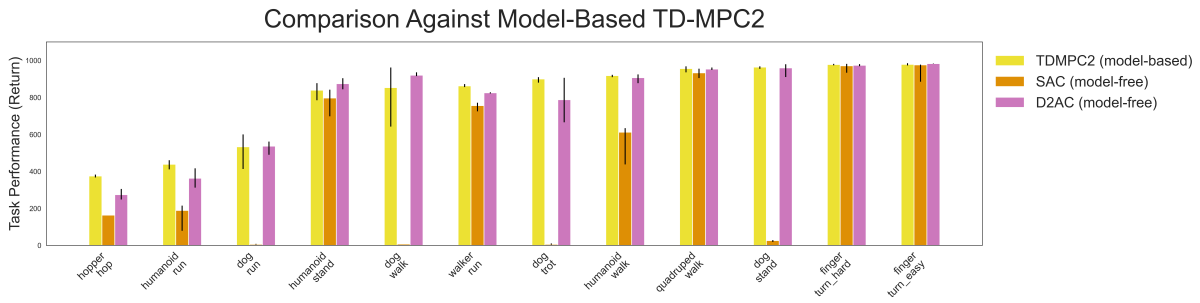

Figure 4: Comparison between model-based TD-MPC2 (Hansen et al., 2023), SAC (Haarnoja et al., 2018), and our method D2AC on DeepMind Control Suite. **D2AC without planning** can achieve results on-par with TD-MPC2.

5.2 Benchmarking on goal-conditioned RL environments

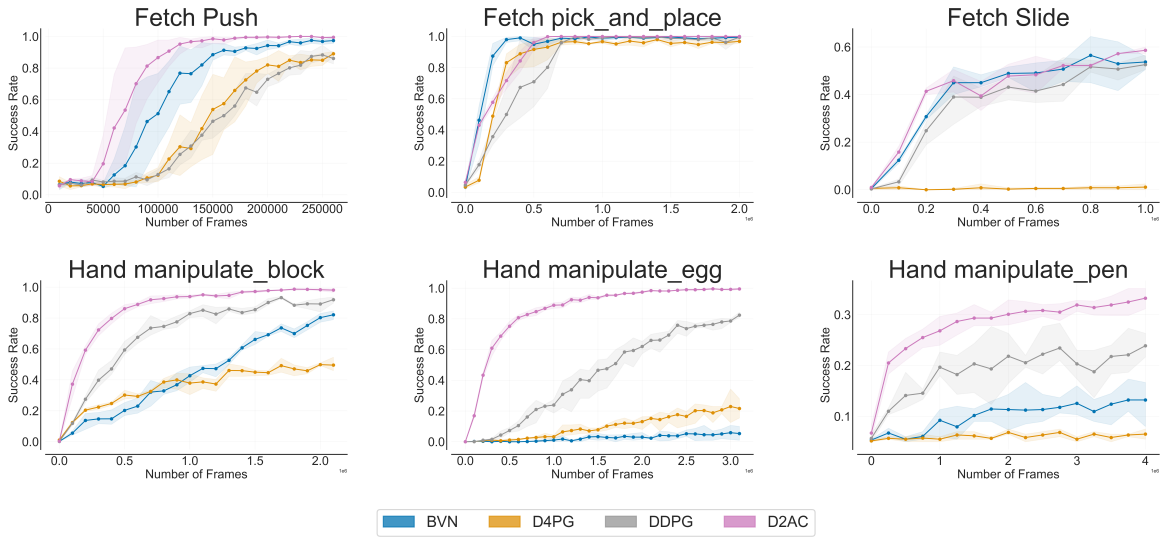

Figure 5: Experiments on **Multi-Goal RL environments with sparse rewards**. Results over 5 seeds.

In Figure 5, we present results on robotic manipulation tasks with sparse rewards. All methods are trained with Hindsight Experience Replay (HER) (Andrychowicz et al., 2017). The baselines include: DDPG (Lillicrap et al., 2015), a distributional version of HER (Eysenbach et al., 2019), and recently proposed Bilinear-Value Network (BVN) (Hong et al., 2022) found to be effective in goal-conditioned RL. We use a centralized learner and 20 sampling workers for all methods. We find that on Fetch tasks, D2AC performs better or on-par with previous SOTA method BVN. On 20-DoF Hand Manipulate tasks, the second-best method is still DDPG, and our method is able to outperform it by a large margin, often both in terms of learning speed and policy performance at convergence. **This performance shows that D2AC is a strong base algorithm that can work out-of-the-box in a variety of settings.**

Interestingly, the previously proposed distributional critic function in (Eysenbach et al., 2019) achieves worst performance than non-distributional critics. We hypothesize that this is because its specialized type of goal-conditioned distributional Bellman backup uses a one-hot target the moment a goal is reached, so it becomes more difficult for the agent to learn how to consistently stay at the goal. Our distributional critics do not make any particular assumptions about the problem structure of goal-conditioned RL, but still achieve very strong results in this setting.

5.3 Biology-Inspired Benchmark: Behavioral Adaptation in Predator–Prey Scenarios

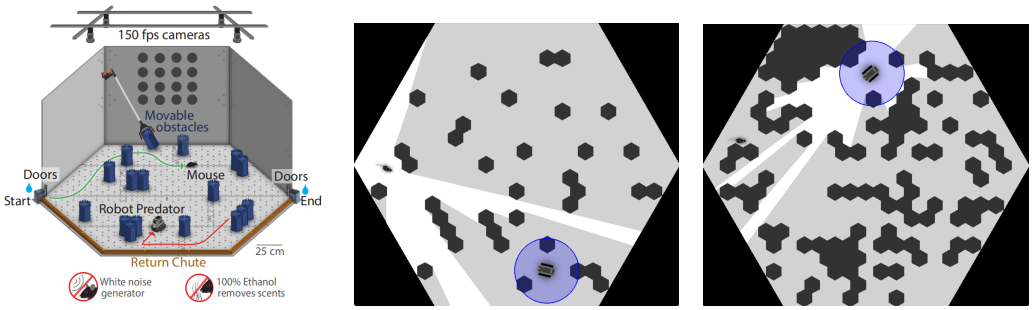

Figure 6: **Left:** Predator–prey arena(hexagonal maze, start to goal, +1 at goal, –1 when predator is within 0.1 units, matching the real-mouse water/air-puff setup). **Middle:** Prey's egocentric view on Map Level 5 (RL sim). **Right:** Predator's egocentric view on Map Level 9 (RL sim).

While standard reinforcement learning benchmarks primarily focus on task completion or reward maximization, they often overlook structural differences in the learned policies. To address this limitation, we introduce a predator–prey environment inspired by biological survival dynamics (Lai et al., 2024). Such scenarios serve as foundational models in both natural and artificial systems for analyzing adaptive decision-making under pressure (Marrow et al., 1996; Han et al., 2025). This domain provides a natural testbed for analyzing the internal structure of learned policies, including diversity and strategic flexibility to adversarial changes, qualities often hidden behind reward curves.

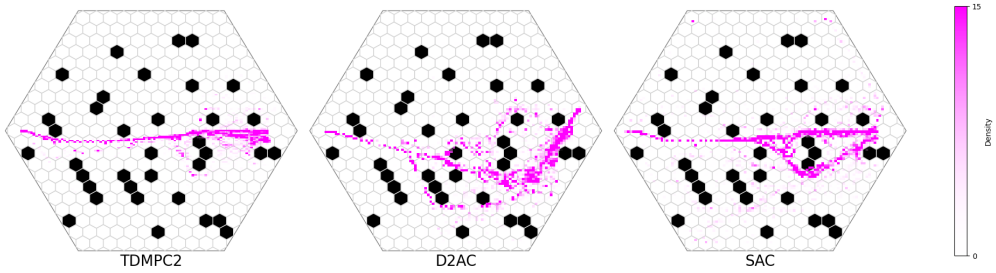

Figure 7: Comparison of state density among TD-MPC2, D2AC, and SAC algorithms in a predator-prey environment. The visualization shows 2000 random episodes for each method, where purple represents the agent's state density as the prey reaches the goal while avoiding predators. D2AC demonstrates superior performance with greater path diversity compared to TD-MPC2 and SAC.

Table 1: Exploration Coverage Analysis over five seeds (values reported as coverage % ± standard deviation)

Algorithm	Visit Coverage ($\tau \geq 1$) [a]	Main Coverage ($\tau \geq 10$) [b]
SAC	35.07 ± 6.07	10.18 ± 1.18
TDMPC	31.32 ± 11.64	10.09 ± 1.76
D2AC	$\mathbf{44.58 \pm 3.16}$	$\mathbf{20.04 \pm 0.49}$

Notes. [a]States visited at least once. [b]States visited at least 10 times.

Metrics: We train three agents (TD-MPC2, SAC, and our proposed D2AC) for 50,000 steps on Map Level 5 and evaluate: (i)*Exploration Coverage* (Table 1) through visit coverage and core-area coverage; (ii)*Survival Rate* over 2,000 episodes; (iii)*Zero-Shot Transfer* to unseen Map Level 9 (Table 2).

Results and Analysis: As shown in Table 1, D2AC achieves the highest visit coverage (44.6%, 9.5% over SAC and 13.2% over TD-MPC2) and doubles TD-MPC2's core coverage, indicating richer exploration. Crucially, this richer exploration directly drives generalization: In zero-shot transfer (Table 2), both TD-MPC2

Table 2: Zero-shot transfer performance from Map Level 5 (trained) to Map Level 9 (unseen). Results are reported as survival rate % ± standard deviation over five seeds.

Algorithm	Level 5 (Trained)	Level 9 (Zero-Shot)	**Change (%)**
TD-MPC2	72.33 ± 1.95	62.35 ± 1.36	−9.98
SAC	86.37 ± 3.23	75.95 ± 7.41	−10.42
D2AC	**87.05 ± 4.24**	**90.69 ± 3.56**	**+3.64**

and SAC degrade by approximately 10%, while D2AC *improves* by 3.6%, demonstrating notable adaptability. Fig. 7 further reveals that TD-MPC2 and SAC exhibit repetitive narrow trajectories, while D2AC explores more diverse paths and dynamically avoids the predator. These findings indicate that our framework not only achieves competitive survival rates, but also uncovers structurally richer strategies.

Why D2AC Excels? We hypothesize that the key to D2AC's success lies in the synergy between its two main components: the distributional critic and the diffusion actor. The critic models the full return distribution over trajectories, capturing higher-order statistics such as variance and skewness, which serve as dense, uncertainty-aware learning signals (Kuang et al., 2023; Lowet et al., 2025). The diffusion actor generates multimodal action proposals by iteratively denoising samples, enabling it to flexibly trade off between risk and reward. By aligning action sampling with the uncertainty contours of the critic's distribution (Jain et al., 2024; Li et al., 2024), D2AC naturally balances exploration and exploitation and produces richer, more informative policy gradients.

5.4 Distributional Critic vs. Diffusion Actor

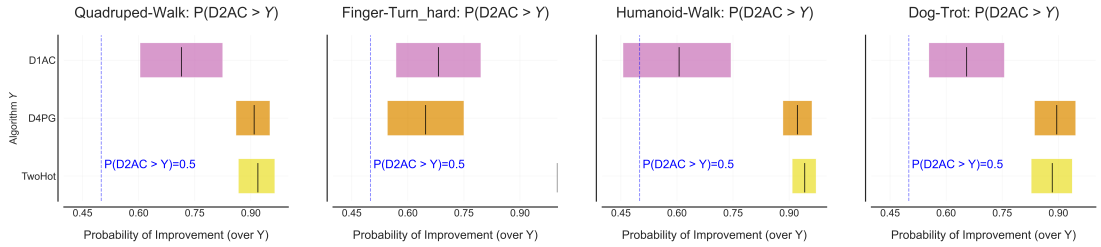

Figure 8: Ablation Studies based on Probability of improvement (Agarwal et al., 2021) using the recommended protocol from rliable. The X-axis gives the probability D2AC improves upon the performance of the baseline algorithm on the Y-axis. The boxes represent 95% confidence intervals for each baseline.

One natural question is the following: what is the relative importance of the diffusion actor vs the distributional critic? Are both components necessary? We test this by introducing an additional method called **D1AC**. This method implements the clipped double distributional critics proposed in our paper, but uses a standard Tanh-Gaussian actor similar to SAC. In Figure 8, we present the results of D1AC alongside D4PG and D2AC to ablate how much which each modification helps. D2AC has a high probability of improvement on a variety of ablation environments considered. Note that P(D2AC > Y) is higher than 0.5 on all choices of Algorithm Y. The improvement of D1AC compared to D4PG is smaller depending on the environment; our hypothesis is that it might be important to increase the representation power of critic and actor at the same time. Furthermore, we hypothesize some environments might be more bottlenecked by the policy rather than the critic function. In addition, the fact that D1AC has much stronger performance than *Two-hot + CDQ* means that combining distributional RL with CDQ is significantly more effective than its counterpart with two-hot discrete critic, which was widely adopted in other methods (Schrittwieser et al., 2020; Hafner et al., 2023).

With respect to the diffusion actor, we are also interested in understanding the relationship between the number of diffusion steps at training vs the final model performance. In particular, we can consider both

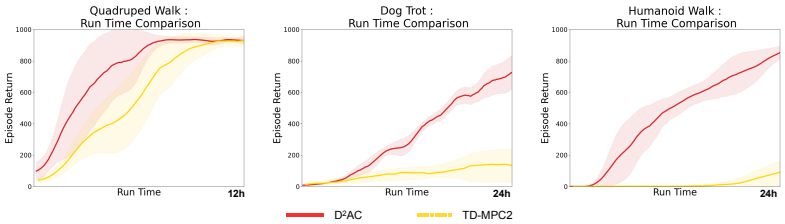

Figure 9: Wall-clock runtime comparison against TD-MPC2 on a single GPU. Thanks to being model-free algorithm without the need for dynamics unrolling and planning, D2AC is a lot faster.

K^{train}, the number of diffusion steps used during training, and K, the number of diffusion steps used at inference. In Table 1, we see that there is some moderate improvements to the final results when training on longer diffusion horizons. Our results agree with the common practice in diffusion models (Song et al., 2020a; Karras et al., 2022) of using more diffusion steps for training and for inference. In addition, we also find that setting $\pi_{\text{ref}} = \pi_\phi$ as done in D2AC policy loss equation 15 rather than just using the actions from the replay buffer $\pi_{\text{ref}} \neq \pi_\phi$ slightly improves results.

Table 3: Performance Statistics at 500K Environment Steps

Configuration	Quadruped Walk	Humanoid Walk
$K^{train} = K = 2$	778.6 ± 227.6	378.7 ± 52.5
$K^{train} = K = 5$	$\mathbf{941.5 \pm 9.7}$	$\mathbf{425.1 \pm 39.9}$
$K^{train} = 5, K = 2$	928.8 ± 15.9	386.9 ± 59.8
$K^{train} = K = 5$ where $\pi_{\text{ref}} \neq \pi_\phi$	829.9 ± 127.5	314.2 ± 76.5

5.5 Limitation of Our Method

Fundamentally, our method is still a model-free RL method; even a more expressive policy class based on denoising score matching cannot fully substitute for the role of planning. Although the denoising diffusion policy improvement we propose already has a lot of similarity to MPPI (Williams et al., 2015a; 2016) in terms of optimization objective, MPPI is explicitly guided by the critic function during the sampling process, and simultaneously considers multiple possible trajectories. We expect that explicit value-based planning should be able to further enhance diffusion performance, but leave it for future work.

6 Conclusion

We have presented D2AC, a model-free RL algorithm that takes inspiration from recent advances in model-based RL (MBRL). By making careful use of a distributional critic and a denoising actor, we are able to capture two salient properties from MBRL algorithms: the ability to optimize over a distribution of possible returns, and the ability to iteratively refine our action selection. Several careful design choices were required to make this algorithm work at its full potential, most notably using a double-Q learning trick with distributional RL and a tailored denoising loss for the actor. The resulting method achieves state-of-the-art model-free performance on both sparse and dense tasks, runs significantly faster than model-based methods, and even excels on a biology-inspired predator–prey benchmark, delivering superior exploration and zero-shot transfer.

There exist a few exciting avenues for future work. D2AC can be plugged into any model-based methods as the model-free RL component; we hope to see how simple planning modules can boost its performance. Our method also did not consider exploration in a direct manner; for some environments, the bottleneck might be exploration. Finally, we are interested in further improving the computational efficiency of D2AC by investigating faster diffusion models (Song et al., 2023; Song & Dhariwal, 2023; Chadebec et al., 2025) and beyond.

References

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. *arXiv preprint arXiv:1806.06920*, 2018.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 34:29304–29320, 2021.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

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

Gabriel Barth-Maron, Matthew W Hoffman, David Budden, Will Dabney, Dan Horgan, Dhruva Tb, Alistair Muldal, Nicolas Heess, and Timothy Lillicrap. Distributed distributional deterministic policy gradients. *arXiv preprint arXiv:1804.08617*, 2018.

Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International conference on machine learning*, pp. 449–458. PMLR, 2017.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*, 2023.

Clement Chadebec, Onur Tasar, Eyal Benaroche, and Benjamin Aubin. Flash diffusion: Accelerating any conditional diffusion model for few steps image generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 15686–15695, 2025.

Yuhui Chen, Haoran Li, and Dongbin Zhao. Boosting continuous control with consistency policy. *arXiv preprint arXiv:2310.06343*, 2023.

Sinho Chewi, Alkis Kalavasis, Anay Mehrotra, and Omar Montasser. Ddpm score matching is asymptotically efficient. In *ICLR 2025 Workshop on Deep Generative Model in Machine Learning: Theory, Principle and Efficacy*.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

Kevin Clark, Paul Vicol, Kevin Swersky, and David J Fleet. Directly fine-tuning diffusion models on differentiable rewards. *arXiv preprint arXiv:2309.17400*, 2023.

Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

Will Dabney, Georg Ostrovski, David Silver, and Rémi Munos. Implicit quantile networks for distributional reinforcement learning. In *International conference on machine learning*, pp. 1096–1105. PMLR, 2018.

Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134:19–67, 2005.

Xiaoyi Dong, Jian Cheng, and Xi Sheryl Zhang. Maximum entropy reinforcement learning with diffusion policy. *arXiv preprint arXiv:2502.11612*, 2025.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, pp. 1407–1416. PMLR, 2018.

Ben Eysenbach, Russ R Salakhutdinov, and Sergey Levine. Search on the replay buffer: Bridging planning and reinforcement learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pp. 1587–1596. PMLR, 2018.

Chen-Xiao Gao, Chenyang Wu, Mingjun Cao, Chenjun Xiao, Yang Yu, and Zongzhang Zhang. Behavior-regularized diffusion policy optimization for offline reinforcement learning. *arXiv preprint arXiv:2502.04778*, 2025.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Shuo Han, German Espinosa, Junda Huang, Daniel A Dombeck, Malcolm A MacIver, and Bradly C Stadie. Of mice and machines: A comparison of learning between real world mice and rl agents. *arXiv preprint arXiv:2505.12204*, 2025.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*, 2022.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023.

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.

Nicolas Heess, Gregory Wayne, David Silver, Timothy Lillicrap, Tom Erez, and Yuval Tassa. Learning continuous control policies by stochastic value gradients. *Advances in neural information processing systems*, 28, 2015.

Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

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

Zhang-Wei Hong, Ge Yang, and Pulkit Agrawal. Bilinear value networks. *arXiv preprint arXiv:2204.13695*, 2022.

Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Mohammadamin Barekatain, Simon Schmitt, and David Silver. Learning and planning in complex action spaces. In *International Conference on Machine Learning*, pp. 4476–4486. PMLR, 2021.

Ehsan Imani and Martha White. Improving regression performance with distributional losses. In *International conference on machine learning*, pp. 2157–2166. PMLR, 2018.

Haque Ishfaq, Guangyuan Wang, Sami Nur Islam, and Doina Precup. Langevin soft actor-critic: Efficient exploration through uncertainty-driven critic learning. *arXiv preprint arXiv:2501.17827*, 2025.

Vineet Jain, Tara Akhound-Sadegh, and Siamak Ravanbakhsh. Sampling from energy-based policies using diffusion. *arXiv preprint arXiv:2410.01312*, 2024.

Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34:1273–1286, 2021.

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pp. 9902–9915. PMLR, 2022.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*, 35:26565–26577, 2022.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Qi Kuang, Zhoufan Zhu, Liwen Zhang, and Fan Zhou. Variance control for distributional reinforcement learning. *arXiv preprint arXiv:2307.16152*, 2023.

Alexander T Lai, German Espinosa, Gabrielle E Wink, Christopher F Angeloni, Daniel A Dombeck, and Malcolm A MacIver. A robot-rodent interaction arena with adjustable spatial complexity for ethologically relevant behavioral studies. *Cell reports*, 43(2), 2024.

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

Steven Li, Rickmer Krohn, Tao Chen, Anurag Ajay, Pulkit Agrawal, and Georgia Chalvatzaki. Learning multimodal behaviors from scratch with diffusion policy gradient. *Advances in Neural Information Processing Systems*, 37:38456–38479, 2024.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Adam S Lowet, Qiao Zheng, Melissa Meng, Sara Matias, Jan Drugowitsch, and Naoshige Uchida. An opponent striatal circuit for distributional reinforcement learning. *Nature*, pp. 1–10, 2025.

Haitong Ma, Tianyi Chen, Kai Wang, Na Li, and Bo Dai. Soft diffusion actor-critic: Efficient online reinforcement learning for diffusion policy. *arXiv preprint arXiv:2502.00361*, 2025.

Paul Marrow, Ulf Dieckmann, and Richard Law. Evolutionary dynamics of predator-prey systems: an ecological perspective. *Journal of mathematical biology*, 34:556–578, 1996.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Anusha Nagabandi, Kurt Konolige, Sergey Levine, and Vikash Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning*, pp. 1101–1112. PMLR, 2020.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pp. 1310–1318. Pmlr, 2013.

Jan Peters and Stefan Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the 24th international conference on Machine learning*, pp. 745–750, 2007.

Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, et al. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*, 2018.

Anil V Rao. A survey of numerical methods for optimal control. *Advances in the astronautical Sciences*, 135 (1):497–528, 2009.

Ralf Römer, Lukas Brunke, Martin Schuck, and Angela P Schoellig. Safe offline reinforcement learning using trajectory-level diffusion models. In *ICRA 2024 Workshop {\textemdash} Back to the Future: Robot Learning Going Probabilistic*, 2024.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

John Schulman, Xi Chen, and Pieter Abbeel. Equivalence between policy gradients and soft q-learning. *arXiv preprint arXiv:1704.06440*, 2017a.

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

Max Schwarzer, Johan Samir Obando Ceron, Aaron Courville, Marc G Bellemare, Rishabh Agarwal, and Pablo Samuel Castro. Bigger, better, faster: Human-level atari with human-level efficiency. In *International Conference on Machine Learning*, pp. 30365–30380. PMLR, 2023.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

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

Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. *arXiv preprint arXiv:2310.14189*, 2023.

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

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Hado Van Hasselt. Double q-learning. *Advances in neural information processing systems*, 23, 2010.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23 (7):1661–1674, 2011.

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. *arXiv preprint arXiv:2311.12908*, 2023.

Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning. *arXiv preprint arXiv:1907.02057*, 2019.

Grady Williams, Andrew Aldrich, and Evangelos Theodorou. Model predictive path integral control using covariance variable importance sampling. *arXiv preprint arXiv:1509.01149*, 2015a.

Grady Williams, Andrew Aldrich, and Evangelos Theodorou. Model predictive path integral control using covariance variable importance sampling. *arXiv preprint arXiv:1509.01149*, 2015b.

Grady Williams, Paul Drews, Brian Goldfain, James M Rehg, and Evangelos A Theodorou. Aggressive driving with model predictive path integral control. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1433–1440. IEEE, 2016.

Long Yang, Zhixiong Huang, Fenghao Lei, Yucun Zhong, Yiming Yang, Cong Fang, Shiting Wen, Binbin Zhou, and Zhouchen Lin. Policy representation via diffusion probability model for reinforcement learning. *arXiv preprint arXiv:2305.13122*, 2023.

Mingqi Yuan, Bo Li, Xin Jin, and Wenjun Zeng. Ultho: Ultra-lightweight yet efficient hyperparameter optimization in deep reinforcement learning. *arXiv preprint arXiv:2503.06101*, 2025.

Lunjun Zhang, Yuwen Xiong, Ze Yang, Sergio Casas, Rui Hu, and Raquel Urtasun. Learning unsupervised world models for autonomous driving via discrete diffusion. *arXiv preprint arXiv:2311.01017*, 2023.

A On Two-hot Functions

The function $h_{\mathbf{z}_q}$ is called the *two-hot* function. Mathematically, $h_{\mathbf{z}_q}(z)$ is defined as:

$$h_{\mathbf{z}_q}(z)[k] = \begin{cases} 1 & z \leq V_{\min} \text{ and } k = 0, \\ 0 & z \leq \mathbf{z}_q[k-1], \\ \frac{z - \mathbf{z}_q[k-1]}{\mathbf{z}_q[k] - \mathbf{z}_q[k-1]} & \mathbf{z}_q[k-1] \leq z \leq \mathbf{z}_q[k], \\ \frac{\mathbf{z}_q[k+1] - z}{\mathbf{z}_q[k+1] - \mathbf{z}_q[k]} & \mathbf{z}_q[k] \leq z \leq \mathbf{z}_q[k+1], \\ 0 & z \geq \mathbf{z}_q[k+1], \\ 1 & z \geq V_{\max} \text{ and } k = N. \end{cases}$$

Which is a piece-wise linear function with the following properties. $\forall z$, if $V_{\min} \leq z \leq V_{\max}$,

$$\sum_{j=0}^{N} h_{\mathbf{z}_q}(z)[j] = 1 \quad \sum_{j=0}^{N} h_{\mathbf{z}_q}(z)[j] \cdot \mathbf{z}_q[j] = z \tag{16}$$

Which means that for any z value, the two-hot function will distribute probability mass in the $[V_{\min}, V_{\max}]$ region that sums up to 1, with a weighted mean equal to z itself. Note that the two-hot function is not the only function that satisfies those properties; for instance, the HL-Gauss parameterization proposed in (Imani & White, 2018) can also satisfy those two conditions.

Lately, a different type of discrete critic based on *two-hot* representation has gained popularly from the success of MuZero (Schrittwieser et al., 2020), Dreamer-v3 (Hafner et al., 2023), and TD-MPC2 (Hansen et al., 2023). To draw the connection between the two types of discrete critics, we first notice the following:

$$\sum_{j=0}^{N} \mathbf{p}[j] = 1 \qquad r + \gamma \sum_{j=0}^{N} \mathbf{z}_q[j] \cdot \mathbf{p}[j] = \sum_{j=0}^{N} (r + \gamma \mathbf{z}_q)[j] \cdot \mathbf{p}[j]$$

It follows that: two-hot discrete critic learning can be reformulated as **simply swapping the order between the sum and the two-hot function in the projection operator of distributional RL**:

$$\mathbf{\Phi}_{\text{dist}}(\mathbf{z}_p, \mathbf{p}, \mathbf{z}_q)[k] = \sum_{j=0}^{N} h_{\mathbf{z}_q}(\mathbf{z}_p[j])[k] \cdot \mathbf{p}[j]$$

$$\mathbf{\Phi}_{\text{twohot}}(\mathbf{z}_p, \mathbf{p}, \mathbf{z}_q)[k] = h_{\mathbf{z}_q}\left(\sum_{j=0}^{N} \mathbf{z}_p[j] \cdot \mathbf{p}[j]\right)[k]$$

And the two-hot label is:

$$\mathbf{a}' \sim \pi_\phi(\cdot \mid \mathbf{s}') \quad \boldsymbol{\ell}_{\text{twohot}} = \mathbf{\Phi}_{\text{twohot}}(r + \gamma \mathbf{z}_q, \mathbf{q}_{\overline{\theta}}(\mathbf{s}', \mathbf{a}'), \mathbf{z}_q)$$
$$L(\theta) = -\boldsymbol{\ell}_{\text{twohot}}^\top \log \mathbf{q}_\theta(\mathbf{s}, \mathbf{a})$$

In comparison, distributional RL can be seen as using soft labels rather than hard labels; it allows the uncertainty about future returns to propagate through Q-learning. Two-hot discrete critic learning does not allow such uncertainty to propagate between different timesteps through Bellman backup.

B Policy Gradients for Variance-Exploding (VE) Diffusion

Let one step of the diffusion process $\mathbf{a}^{(k)} \to \mathbf{a}^{(k-1)}$ for policy π_ϕ be:

$$\pi_\phi(\mathbf{a}^{(k-1)} \mid \mathbf{a}^{(k)}, \mathbf{s}) = \mathcal{N}(\mathbf{a}^{(k-1)} \mid \boldsymbol{\mu}_\phi(\mathbf{a}^{(k)}, k; \mathbf{s}), (\sigma_k^2 - \sigma_{k-1}^2)\mathbf{I})$$
$$\boldsymbol{\mu}_\phi(\mathbf{a}^{(k)}, k; \mathbf{s}) = \mathbf{a}^{(k)} + (\sigma_k^2 - \sigma_{k-1}^2)\nabla_{\mathbf{a}} \log p(\mathbf{a}^{(k)} \mid \sigma_k, \mathbf{s})$$
$$\nabla_{\mathbf{a}} \log p(\mathbf{a} \mid \sigma, \mathbf{s}) = (D_\phi(\mathbf{a}, \sigma; \mathbf{s}) - \mathbf{a})/\sigma^2$$

where we can see that

$$\begin{aligned}\boldsymbol{\mu}_\phi(\mathbf{a}^{(k)}, k; \mathbf{s}) &= \mathbf{a}^{(k)} + \frac{(\sigma_k^2 - \sigma_{k-1}^2)}{\sigma_k^2}\Big(D_\phi(\mathbf{a}^{(k)}, \sigma_k; \mathbf{s}) - \mathbf{a}^{(k)}\Big) \\ &= \frac{\sigma_{k-1}^2}{\sigma_k^2}\mathbf{a}^{(k)} + \frac{(\sigma_k^2 - \sigma_{k-1}^2)}{\sigma_k^2} D_\phi(\mathbf{a}^{(k)}, \sigma_k; \mathbf{s})\end{aligned}$$

In addition, let the latent variables $\mathbf{a}^{(1)} \cdots \mathbf{a}^{(K)}$ be:

$$q(\mathbf{a}^{(k)} \mid \mathbf{a}) = \mathcal{N}(\mathbf{a}^{(k)} \mid \mathbf{a}, \sigma_k^2 \mathbf{I})$$

Then we have the marginal distribution denoted as:

$$p(\mathbf{a}^{(k)}) = \int p_{\text{data}}(\mathbf{a}) q(\mathbf{a}^{(k)} \mid \mathbf{a}) \mathrm{d}\mathbf{a}$$

Under the assumption that $\sigma_{\max}$ is sufficiently large, the initial noise distribution is pre-defined as:

$$p(\mathbf{x}^{(K)}) = \mathcal{N}(\mathbf{0}, \sigma_{\max}^2 \mathbf{I})$$

The full policy distribution is:

$$\pi_\phi(\mathbf{a} \mid \mathbf{s}) = \int p(\mathbf{a}^{(K)}) \prod_{k=1}^{K} \pi_\phi(\mathbf{a}^{(k-1)} \mid \mathbf{s}, \mathbf{a}^{(k)}) \mathrm{d}\mathbf{a}_{1:K}$$

The distribution of $\mathbf{a}^{(k)}$ given $\mathbf{a}^{(i)}$, for $k > i$, is given by:

$$q(\mathbf{a}^{(k)} \mid \mathbf{a}^{(i)}) = \mathcal{N}(\cdot \mid \mathbf{a}^{(i)}, (\sigma_k^2 - \sigma_i^2)\mathbf{I})$$

Since the forward process is Markov, we have

$$q(\mathbf{a}^{(k)}, \mathbf{a}^{(i)} \mid \mathbf{a}) = q(\mathbf{a}^{(k)} \mid \mathbf{a}) \cdot q(\mathbf{a}^{(i)} \mid \mathbf{a})$$

The posterior distribution can be written as:

$$q(\mathbf{a}^{(i)} \mid \mathbf{a}^{(k)}, \mathbf{a})$$

Which can be computed in closed-form using the conjugate prior of Gaussian distributions:

$$q(\mathbf{a}^{(i)} \mid \mathbf{a}^{(k)}, \mathbf{a}) = \mathcal{N}\bigg(\cdot \,\Big|\, \frac{\sigma_i^2}{\sigma_k^2}\mathbf{a}^{(k)} + \frac{(\sigma_k^2 - \sigma_i^2)}{\sigma_k^2}\mathbf{a}, \sigma_i^2 \frac{(\sigma_k^2 - \sigma_i^2)}{\sigma_k^2} \bigg)$$

Then Jensen's inequality gives us the following lower bound (Sohl-Dickstein et al., 2015; Ho et al., 2020):

$$\begin{aligned} -\log \pi_\phi(\mathbf{a} \mid \mathbf{s}) &\leq L_{\text{prior}} + \sum_{k=1}^{K} \mathbb{E}_{q(\mathbf{a}^{(k)}|\mathbf{a})} D_{KL}\Big[q(\mathbf{a}^{(k-1)}|\mathbf{a}^{(k)}, \mathbf{a}) \parallel \pi_\phi(\mathbf{a}^{(k-1)}|\mathbf{a}^{(k)}, \mathbf{s}) \Big] \\ L_{\text{prior}} &= D_{KL}(q(\mathbf{a}^{(K)} \mid \mathbf{a}) \parallel p(\mathbf{a}^{(K)})) \end{aligned}$$

From the result of (Kingma et al., 2021), we know that

$$\begin{aligned} D_{KL}(q(\mathbf{a}^{(k-1)}|\mathbf{a}^{(k)}, \mathbf{a}) \parallel \pi_\phi(\mathbf{a}^{(k-1)}|\mathbf{a}^{(k)}, \mathbf{s})) &= \frac{1}{2\sigma_{k-1}^2} \frac{\sigma_k^2 - \sigma_{k-1}^2}{\sigma_k^2} \|\mathbf{a} - D_\phi(\mathbf{a}^{(k)}, \sigma_k; \mathbf{s})\|^2 \\ &= \frac{1}{2}\left(\frac{1}{\sigma_{k-1}^2} - \frac{1}{\sigma_k^2} \right) \|\mathbf{a} - D_\phi(\mathbf{a}^{(k)}, \sigma_k; \mathbf{s})\|^2 \end{aligned}$$

Allowing us to arrive at the following result:

$$\log \pi_\phi(\mathbf{a}|\mathbf{s}) \exp Q(\mathbf{s}, \mathbf{a}) \geq -\mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}), k \sim \text{Unif}(1 \cdots K)} [\lambda_{\text{pg}}(k) \|\mathbf{a} - D_\phi(\mathbf{a} + \sigma_k \boldsymbol{\epsilon}, \sigma_k; \mathbf{s})\|^2 \exp Q(\mathbf{s}, \mathbf{a})]$$

where the weightings for policy gradient objective λ_{pg} are:

$$\lambda_{\text{pg}}(k) = (1/\sigma_{k-1}^2 - 1/\sigma_k^2) \cdot (K/2)$$

C Proof of Proposition 1

Proposition 1. *Define the data distribution under diffusion step $k \geq 0$ to be (with $\hat{\sigma} \geq \sigma_{min}$):*

$$p_k(\mathbf{a}|\mathbf{s}) = \int \pi(\mathbf{a}^{(0)}|\mathbf{s}) \mathcal{N}(\mathbf{a}|\mathbf{a}^{(0)}, \sigma_k^2 \mathbf{I}) \mathrm{d}\mathbf{a}^{(0)} \tag{17}$$

$$\hat{p}_k(\mathbf{a}|\mathbf{s}) = \int p_{k+1}(\mathbf{u}|\mathbf{s}) \mathcal{N}(\mathbf{a}|D(\mathbf{u}, \sigma_{k+1}; \mathbf{s}), \hat{\sigma}^2 \mathbf{I}) \mathrm{d}\mathbf{u} \tag{18}$$

Under the assumptions that (i) entropy of p_k and $\hat{p}_k$ strictly increases with k; (ii) the KL distance from p_k and $\hat{p}_k$ to optimal policy p^* strictly decreases with k; (iii) $\hat{\sigma}$ is sufficiently large such that $D_{KL}(\hat{p}_0 \parallel p^*) \geq D_{KL}(p_0 \parallel p^*)$. Then for any state $\mathbf{s}$ and diffusion step k, we have

$$\mathbb{E}_{p_0}[Q] \geq \mathbb{E}_{\hat{p}_k}[Q]$$

Proof. Fix an arbitrary state $s \in \mathcal{S}$. Because the Boltzmann optimal policy satisfies

$$p^*(a \mid s) = \frac{\exp\{Q(s,a)\}}{Z(s)}, \qquad Z(s) = \int \exp\{Q(s,a')\}\, da',$$

it follows that $Q(s,a) = \log Z(s) - \log p^*(a \mid s)$. For any distribution $q(\cdot \mid s)$ this identity yields

$$\mathbb{E}_q[Q] = \log Z(s) - D_{\mathrm{KL}}\big(q \,\|\, p^*\big) - \mathcal{H}(q), \tag{19}$$

where $\mathcal{H}(\cdot)$ denotes differential entropy. Applying equation 19 to $q = p_0$ and $q = \hat{p}_k$ and taking their difference gives

$$\mathbb{E}_{p_0}[Q] - \mathbb{E}_{\hat{p}_k}[Q] = \underbrace{D_{\mathrm{KL}}\big(\hat{p}_k \,\|\, p^*\big) - D_{\mathrm{KL}}\big(p_0 \,\|\, p^*\big)}_{:=\Delta_{\mathrm{KL}}(k)} + \underbrace{\mathcal{H}(\hat{p}_k) - \mathcal{H}(p_0)}_{:=\Delta_H(k)}. \tag{20}$$

KL term. Assumption (ii) states that $k \mapsto D_{\mathrm{KL}}\big(\hat{p}_k \| p^*\big)$ is strictly decreasing, and Assumption (iii) gives $D_{\mathrm{KL}}\big(\hat{p}_0 \| p^*\big) \geq D_{\mathrm{KL}}\big(p_0 \| p^*\big)$. Therefore $\Delta_{\mathrm{KL}}(k) \geq 0$ for every $k \geq 0$.

Entropy term. Assumption (i) ensures $\mathcal{H}(\hat{p}_k) \geq \mathcal{H}(\hat{p}_0)$, so it remains to verify $\mathcal{H}(\hat{p}_0) \geq \mathcal{H}(p_0)$. Let $A^{(0)} \sim \pi(\cdot \mid s)$ and draw independent Gaussian noises $\varepsilon_0 \sim \mathcal{N}(0, \sigma_1^2 I)$ and $\varepsilon_1 \sim \mathcal{N}(0, \hat{\sigma}^2 I)$. Then $A^{(0)} + \varepsilon_0 \sim p_1$, while

$$Y := D\big(A^{(0)} + \varepsilon_0, \sigma_1; s\big) + \varepsilon_1 \sim \hat{p}_0.$$

For independent random vectors X and Z, Lemma 17.2.1 of Cover (1999) asserts that

$$\mathcal{H}(X + Z) \;\geq\; \mathcal{H}(X), \tag{21}$$

with strict inequality whenever Z is non-degenerate and X is not already Gaussian with the same covariance. Applying equation 21 to $X = D\big(A^{(0)} + \varepsilon_0, \sigma_1; s\big)$ and $Z = \varepsilon_1$ gives

$$\mathcal{H}(\hat{p}_0) = \mathcal{H}(Y) \;\geq\; \mathcal{H}\big(D(A^{(0)} + \varepsilon_0, \sigma_1; s)\big) \;\geq\; \mathcal{H}(p_1),$$

where the final inequality follows because deterministic mappings cannot raise differential entropy. Assumption (i) further yields $\mathcal{H}(p_1) > \mathcal{H}(p_0)$, hence

$$\mathcal{H}(\hat{p}_0) \;\geq\; \mathcal{H}(p_0), \quad \text{so} \quad \Delta_H(k) \;\geq\; 0 \;\; \forall k \geq 0. \tag{22}$$

Conclusion. Both terms on the right-hand side of equation 20 are non-negative; consequently

$$\mathbb{E}_{p_0}[Q] - \mathbb{E}_{\hat{p}_k}[Q] \;\geq 0 \quad \Longrightarrow \quad \mathbb{E}_{p_0}[Q] \;\geq\; \mathbb{E}_{\hat{p}_k}[Q], \qquad \forall k \geq 0.$$

□

D MPPI

Model Predictive Path Integral Control (MPPI) applies an iterative process of sampling and return-weighted refinement (Williams et al., 2015a; 2016; 2015b; Nagabandi et al., 2020; Hansen et al., 2022). Initial trajectories are sampled from a noise distribution:

$$\begin{aligned} \mathbf{x}_{i=0}^{k=0\cdots K-1} &\sim \mathcal{N}(\cdot \mid \boldsymbol{\mu}_{i=0}, \boldsymbol{\sigma}_{i=0}^2) \\ \boldsymbol{\mu}_{i=0} &= \mathbf{0} \\ \boldsymbol{\sigma}_{i=0}^2 &= \sigma_{\max}^2 \mathbf{I} \end{aligned}$$

At the i-th iteration, MPPI samples K action sequences $\mathbf{x}_i^{k=0\cdots K-1}$ from

$$\mathcal{N}(\cdot \mid \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i^2)$$

uses stochastic (virtual) rollouts to estimate the K empirical returns of each action sequences $\{R\}_{k=0}^{K-1}$. It then updates the sampling distribution according to

$$\boldsymbol{\mu}_{i+1} = \left(\sum_{k=0}^{K-1} \exp(R_k/\lambda)\mathbf{x}_i^k\right) \Big/ \left(\sum_{k=0}^{K-1} \exp(R_k/\lambda)\right)$$
$$\boldsymbol{\sigma}_{i+1}^2 = \left(\sum_{k=0}^{K-1} \exp(R_k/\lambda)(\mathbf{x}_i^k - \boldsymbol{\mu}_{i+1})^2\right) \Big/ \left(\sum_{k=0}^{K-1} \exp(R_k/\lambda)\right)$$

MPPI uses a distribution of returns to update the action proposals to higher-return regions.

E On Tanh Squashing of Actions

The most commonly used action distribution for continuous control is a Gaussian distribution with tanh squashing (Haarnoja et al., 2018). The policy π_ϕ outputs the mean $\boldsymbol{\mu}_\phi$ and log sigma, which we exponentiate to get $\boldsymbol{\sigma}_\phi$.

$$\mathbf{u} \sim \mathcal{N}(\cdot \mid \boldsymbol{\mu}_\phi(\mathbf{s}), \boldsymbol{\sigma}_\phi^2(\mathbf{s})) \quad \mathbf{a} = \tanh(\mathbf{u}) \tag{23}$$

where $\mathbf{u}$ is sampled using the reparameterization trick (Kingma & Welling, 2013): $\mathbf{u} = \boldsymbol{\mu}_\phi + \boldsymbol{\epsilon} \odot \boldsymbol{\sigma}_\phi, \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$.

Applying the change of variable formula, The change of variable formula says

$$\log \pi(\mathbf{a} \mid \mathbf{s}) = \log \mathcal{N}(\mathbf{u} \mid \boldsymbol{\mu}_\phi, \boldsymbol{\sigma}_\phi^2) - \sum_i \log(1 - \tanh^2(\mathbf{u}_i))$$

And the second part can be written as:

$$\begin{aligned}
\log(1 - \tanh^2(\mathbf{u}))) &= 2 \log \mathbf{u} \\
&= 2 \log 2 - 2 \log(e^{\mathbf{u}} + e^{-\mathbf{u}}) \\
&= 2 \log 2 - 2 \log(e^{\mathbf{u}}(1 + e^{-2\mathbf{u}})) \\
&= 2 \log 2 - 2\mathbf{u} - 2 \log(1 + e^{-2\mathbf{u}}) \\
&= 2 \log 2 - 2\mathbf{u} - 2\zeta(-2\mathbf{u}) \\
&= 2[\log 2 - \mathbf{u} - \zeta(-2\mathbf{u})]
\end{aligned}$$

Thus, we use an easy-to-compute and numerically stable form of the log likelihood of Tanh Gaussian distribution:

$$f^{\text{ent}}(\mathbf{u}, \boldsymbol{\mu}, \boldsymbol{\sigma}) = \log \mathcal{N}(\mathbf{u} \mid \boldsymbol{\mu}, \boldsymbol{\sigma}^2) - 2 \cdot \mathbf{1}^\top [\log 2 - \mathbf{u} - \zeta(-2\mathbf{u})] \tag{24}$$

Thus, we can use the squashing to enforce the action boundary, and apply entropy regularization (Haarnoja et al., 2018):

$$L(\alpha) = -\alpha[f_\phi^{\text{ent}} - \lambda_{\text{ent}}|\mathcal{A}|]$$

Which we apply to policy network similar to other model-free baselines.

F Policy Network Parameterization

Let $p_\sigma(\tilde{\mathbf{u}}) = \int p_{\text{data}}(\mathbf{u}) \mathcal{N}(\tilde{\mathbf{u}} \mid \mathbf{u}, \sigma^2 \mathbf{I}) \mathrm{d}\mathbf{u}$. The min $\sigma_{\min}$ and max $\sigma_{\max}$ of noise levels are selected such that $p_{\sigma_{\min}}(\tilde{\mathbf{u}}) \approx p_{\text{data}}(\mathbf{u})$ and $p_{\sigma_{\max}}(\tilde{\mathbf{u}}) \approx \mathcal{N}(\mathbf{0}, \sigma_{\max}^2 \mathbf{I})$. We adopt the EDM denoiser parameterization (Karras et al., 2022):

$$\begin{aligned}
\boldsymbol{\mu}_\phi(\mathbf{s}, \mathbf{u}, \sigma) = c_{\text{skip}}(\sigma)\mathbf{u} \\
+ c_{\text{out}}(\sigma) F_\phi(\mathbf{s}, c_{\text{in}}(\sigma)\mathbf{u}, c_{\text{noise}}(\sigma))
\end{aligned} \tag{25}$$

Algorithm 1 Policy Optimization (*under Tanh Action Squashing*)

procedure UPDATE($\phi, \alpha \mid \mathbf{s}, \mathbf{a}, \sigma, Q$)
Input: policy parameters ϕ, entropy coefficient α.
Input: state $\mathbf{s}$, continuous action $\mathbf{a} \in (-\mathbf{1}, \mathbf{1})$.
Input: differentiable $Q(\mathbf{s}, \mathbf{a})$, noise level σ.
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}), \quad \tilde{\mathbf{u}} = \tanh^{-1}(\mathbf{a}) + \boldsymbol{\epsilon} \cdot \sigma$
$\hat{\boldsymbol{\mu}}_\phi = \boldsymbol{\mu}_\phi(\mathbf{s}, \tilde{\mathbf{u}}, \sigma), \quad \hat{\boldsymbol{\sigma}}_\phi = \boldsymbol{\sigma}_\phi(\mathbf{s}, \tilde{\mathbf{u}}, \sigma)$
$\mathbf{x}_\phi \sim \mathcal{N}(\cdot \mid \hat{\boldsymbol{\mu}}_\phi, \hat{\boldsymbol{\sigma}}_\phi^2)$
$\mathbf{x}^{\text{target}} = \mathbf{x} + \nabla_{\mathbf{x}} Q(\mathbf{s}, \tanh(\mathbf{x}))\big|_{\mathbf{x}=\mathbf{x}_\phi}$ ▷ Eq equation 15
$\hat{f}_\phi^{\text{ent}} = f^{\text{ent}}(\mathbf{x}_\phi, \hat{\boldsymbol{\mu}}_\phi, \hat{\boldsymbol{\sigma}}_\phi)$ ▷ Eq equation 24
$\phi \leftarrow \text{SGD}(\phi, \nabla_\phi[\|\mathbf{x}_\phi - \mathbf{x}^{\text{target}}\|^2 + \alpha \hat{f}_\phi^{\text{ent}}])$
$\alpha \leftarrow \text{SGD}(\alpha, \nabla_\alpha[-\alpha(\hat{f}_\phi^{\text{ent}} - \lambda_{\text{ent}}|\mathcal{A}|)])$ ▷ Update temperature
end procedure

Algorithm 2 : **D2 Actor Critic**

procedure ACTORCRITIC($\mathcal{D}, \theta, \phi, \alpha$)
Input: $(\mathbf{s}, \mathbf{a}, \mathbf{s}', r) \sim \mathcal{D}$
Input: critic parameters θ, policy parameters ϕ.
Input: entropy coefficient α.
$\mathbf{a}' \sim \pi_\phi(\cdot \mid \mathbf{s}')$ ▷ Diffusion Sampling
Optimize θ with clipped double distributional RL equation **??**
$i \sim \text{Unif}\{1 \cdots K^{\text{train}}\}, \quad j \sim \text{Unif}\{1 \cdots K\}$
$\sigma_i \leftarrow \sigma(\frac{i-1}{K^{\text{train}}-1}), \quad \sigma_j \leftarrow \sigma(\frac{j-1}{K-1})$ ▷ Select noise levels in EDM
Update ϕ, α on $(\mathbf{s}, \mathbf{a}, \sigma_i)$ and $(\mathbf{s}', \mathbf{a}', \sigma_j)$. ▷ Alg 1
Target network update: $\overline{\theta} \leftarrow \tau\overline{\theta} + (1-\tau)\theta$ ▷ Exponential moving average.
end procedure

With the specific functions being:

$$
\begin{aligned}
c_{\text{skip}}(\sigma) &= \sigma_{\text{data}}^2/(\sigma^2 + \sigma_{\text{data}}^2) \\
c_{\text{out}}(\sigma) &= \sigma \cdot \sigma_{\text{data}}/\sqrt{\sigma^2 + \sigma_{\text{data}}^2} \\
c_{\text{in}}(\sigma) &= 1/\sqrt{\sigma^2 + \sigma_{\text{data}}^2} \\
c_{\text{noise}}(\sigma) &= \log \sigma
\end{aligned}
$$

This parameterization offers powerful inductive bias for an iterative denoising process that has been shown to work well. $\boldsymbol{\sigma}_\phi(\mathbf{s}, \mathbf{a}, \sigma)$ uses the same network F_ϕ except for the last linear layer. The conditioning of $c_{\text{noise}}(\sigma)$ is done via positional embedding (Vaswani et al., 2017).

A sequence of noise levels is used $\sigma_{\min} = \sigma_1 < \sigma_2 < \cdots < \sigma_M = \sigma_{\max}$. We follow EDM in setting $\sigma_i = \sigma(\frac{i-1}{M-1})$ for $i = \{1, \cdots, M\}$ and $\rho = 7$:

$$
\sigma(\eta) = \left(\sigma_{\min}^{1/\rho} + \eta\left(\sigma_{\max}^{1/\rho} - \sigma_{\min}^{1/\rho}\right)\right)^\rho \tag{26}
$$

G Hyperparameters

Table 4: Hyperparameters for D2AC

Parameters	Value
Batch size	256
Optimizer	AdamW adamw
Learning rate for policy	0.001
Learning rate for critic	0.001
Learning rate for temperature α	0.0001
α initialization	0.2
Weight Decay	0.0001
Number of hidden layers (all networks)	2
Number of hidden units per layer	256
Non-linearity	Layer Normalization (Ba et al., 2016) + ReLU
γ	0.99
λ_{ent}	0.0
Polyak for target network	0.995
Target network update interval	1 for dense rewards and predator-prey 10 for sparse rewards
Ratio between env vs optimization steps	1 for dense rewards and predator-prey 2 for sparse rewards
Initial random trajectories	200
Number of parallel workers	4 for dense rewards and predator-prey 20 for sparse rewards
Update every number of steps in environment	same as Number of parallel workers
Replay buffer size	10^6 for dense rewards and predator-prey 2.5×10^6 for sparse rewards
$[V^{\min}, V^{\max}]$	$[-1000, 1000]$ for DM control $[-200, 200]$ for predator-prey $[-50, 0]$ for Multi-Goal RL
Number of bins (size of the support $\mathbf{z}_q$)	201 for DM control 201 for predator-prey 101 for Multi-Goal RL
$\sigma_{\min}$	0.05
$\sigma_{\max}$	2.0
σ_{data}	1.0
ρ	7
M	2
M^{train}	5
Noise-level conditioning	Position encoding on $(10^3 \log \sigma)/4$
Embedding size for noise-level conditioning	32