Locally Persistent Exploration in Continuous Control Tasks with Sparse Rewards

Susan Amin [*1 2] **Maziar Gomrokchi** [*1 2] **Hossein Aboutalebi** [*3 4] **Harsh Sajita** [1 2] **Doina Precup** [1 2]

Abstract

A major challenge in reinforcement learning is the design of exploration strategies, especially for environments with sparse reward structures and continuous state and action spaces. Intuitively, if the reinforcement signal is very scarce, the agent should rely on some form of short-term memory in order to cover its environment efficiently. We propose a new exploration method, based on two intuitions: (1) the choice of the next exploratory action should depend not only on the (Markovian) state of the environment, but also on the agent's trajectory so far, and (2) the agent should utilize a measure of spread in the state space to avoid getting stuck in a small region. Our method leverages concepts often used in statistical physics to provide explanations for the behavior of simplified (polymer) chains in order to generate persistent (locally self-avoiding) trajectories in state space. We discuss the theoretical properties of locally self-avoiding walks and their ability to provide a kind of short-term memory through a decaying temporal correlation within the trajectory. We provide empirical evaluations of our approach in a simulated 2D navigation task, as well as higher-dimensional MuJoCo continuous control locomotion tasks with sparse rewards.

1. Introduction

As reinforcement learning agents typically learn tasks through interacting with the environment and receiving reinforcement signals, a fundamental problem arises when these signals are rarely available. The sparsely distributed rewards call for a clever exploration strategy that exposes the agent to the unseen regions of the space via keeping track of the visited state-action pairs (Fu et al., 2017; Nair et al., 2018). However, that cannot be the case for high-dimensional continuous space-and-action spaces, as defining a notion of density for such tasks is intractable and heavily task-dependent (Andrychowicz et al., 2017; Taïga et al., 2019).

Here, we introduce an exploration algorithm that works independently of the extrinsic rewards received from the environment and is inherently compatible with continuous state-and-action tasks. Our proposed approach takes into account the agent's short-term memory regarding the action trajectory, as well as the trajectory of the observed states in order to sample the next exploratory action. The main intuition is that in a pure exploration mode with minimal extrinsic reinforcement, the agent should plan trajectories that expand in the available space and avoid getting stuck in small regions. In other words, the agent may need to be "persistent" in its choice of actions; for example, in a locomotion task, an agent may want to pick a certain direction and maintain it for some number of steps in order to ensure that it can move away from its current location, where it might be stuck at. The second intuition is that satisfying the first condition requires a notion of spread measure in the state space to warrant the agent's exposure to unvisited regions. Moreover, in sparse reward settings, while the agent's primary intention must be to avoid being trapped in local regions by maintaining a form of short-term memory, it must still employ a form of memory evaporation mechanism to maintain the possibility of revisiting the informative states. Note that in continuous state-and-action settings, modern exploration methods (Ostrovski et al., 2017; Houthooft et al., 2016b; Ciosek et al., 2019) fail to address the fore-mentioned details simultaneously.

Our polymer-based exploration technique (PolyRL) is inspired by the theory of freely-rotating chains (FRCs) in polymer physics to implement the aforementioned intuitions. FRCs describe the chains (collections of transitions or moves) whose successive segments are correlated in their orientation. This feature introduces a finite (short-term) stiffness (persistence) in the chain, which induces what we call *locally* self-avoiding random walks (LSA-RWs). The strategy that emerges from PolyRL induces consistent move-

[*]Equal contribution [1]Department of Computer Science, McGill University, Montréal, Québec, Canada [2]Mila - Québec Artificial Intelligence Institute, Montréal, Québec, Canada [3]Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada [4]Waterloo Artificial Intelligence Institute, University of Waterloo, Waterloo, Ontario, Canada. Correspondence to: Susan Amin <susan.amin@mail.mcgill.ca>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

ment, without the need for exact action repeats (*e.g.* methods suggested by (Dabney et al., 2020; Lakshminarayanan et al., 2017; Sharma et al., 2017)), and can maintain the rigidity of the chain as required. Moreover, unlike action-repeat strategies, PolyRL is inherently applicable in continuous action-state spaces without the need to use any discrete representation of action or state space. The local self-avoidance property in a PolyRL trajectory cultivates an orientationally persistent move in the space while maintaining the possibility of revisiting different regions in space.

To construct LSA-RWs, PolyRL selects persistent actions in the action space and utilizes a measure of spread in the state space, called the *radius of gyration*, to maintain the (orientational) persistence in the chain of visited states. The PolyRL agent breaks the chain and performs greedy action selection once the correlation between the visited states breaks. The next few exploratory actions that follow afterward, in fact, act as a perturbation to the last greedy action, which consequently preserves the orientation of the greedy action. This feature becomes explicitly influential after the agent is exposed to some reinforcement, and the policy is updated, as the greedy action guides the agent's movement through the succeeding exploratory chain of actions.

2. Notation and Problem Formulation

We consider the usual MDP setting, in which an agent interacts with a Markov Decision Process $\langle \mathcal{S}, \mathcal{A}, P, r \rangle$, where $\mathcal{S} \subseteq \mathbb{R}^{d_\mathcal{S}}$ and $\mathcal{A} \subseteq \mathbb{R}^{d_\mathcal{A}}$ are continuous state and action spaces, respectively; $P : \mathcal{S} \times \mathcal{A} \to (\mathcal{S} \to [0, 1])$ represents the transition probability kernel, and $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function. Moreover, we make a smoothness assumption on P,

Assumption 1 *The transition probability kernel P is Lipschitz w.r.t. its action variable, in the sense that there exists $C > 0$ such that for all $(s, a, a') \in \mathcal{S} \times \mathcal{A} \times \mathcal{A}$ and measurable set $\mathcal{B} \subset \mathcal{S}$,*

$$|P(\mathcal{B}|s, a) - P(\mathcal{B}|s, a')| \leq C\|a - a'\|. \tag{1}$$

Assumption 1 has been used in the literature for learning in domains with continuous state-action spaces (Antos et al., 2008), as the assumption on the smoothness of MDP becomes crucial in such environments (Antos et al., 2008; Bartlett & Tewari, 2007; Ng & Jordan, 2000). Note that we only use the Lipschitz smoothness assumption on P for the theoretical convenience, and we later provide experimental results in environments that do not satisfy this assumption. Furthermore, we assume that the state and action spaces are inner product spaces.

We show the trajectory of selected exploratory actions in the action space $\mathcal{A}$ as $\tau_\mathcal{A} = (a_0, ..., a_{T_e-1})$ and the trajectory of the visited states in the state space $\mathcal{S}$ as $\tau_\mathcal{S} = (s_0, ..., s_{T_e-1})$, where T_e denotes the number of exploratory time steps in a piece-wise persistent trajectory, and is reset after each exploitation step. Moreover, we define

$$\Omega(\tau_\mathcal{S}, \tau_\mathcal{A}) := \{s \in \mathcal{S} | \Pr[S_{T_e} = s | s_{T_e-1}, a_{T_e-1}] > 0\} \tag{2}$$

as the set of probable states $s \in \mathcal{S}$ observed at time T_e given s_{T_e-1} from $\tau_\mathcal{S}$ and the selected action a_{T_e-1} from $\tau_\mathcal{A}$. For simplicity, in the rest of this manuscript, we denote $\Omega(\tau_\mathcal{S}, \tau_\mathcal{A})$ by Ω. In addition, the concatenation of the trajectory $\tau_\mathcal{S}$ and the state s visited at time step T_e is denoted as the trajectory $\tau'_\mathcal{S} := (\tau_\mathcal{S}, s)$. For the theoretical analysis purposes, and in order to show the expansion of the visited-states trajectory in the state space, we choose to transform $\tau_\mathcal{S}$ into a sequence of vectors connecting every two consecutive states (bond vectors),

$$\omega_{\tau_\mathcal{S}} = \{\omega_i\}_{i=1}^{T_e-1}, \quad where \quad \omega_i = s_i - s_{i-1}. \tag{3}$$

Finally, we define Self-Avoiding Random Walks (LSA-RWs), inspired by the theory of freely-rotating chains (FRCs), where the correlation between $|i - j|$ consecutive bond vectors is a decaying exponential with the correlation number Lp (persistence number). Lp represents the number of time steps, after which the bond vectors forget their initial orientation.

Definition 1 Locally Self-Avoiding Random Walks (LSA-RWs) *A sequence of random bond vectors $\omega = \{w_i\}_{i=1}^{T_e}$ is Locally Self-Avoiding with persistence number $Lp > 1$, if there exists $b_o > 0$ for all $i, j \in [T_e]$ such that, 1) $\mathbb{E}[\|w_i\|^2] = b_o^2$ and 2) $\mathbb{E}[w_i.w_j] \sim b_o^2 e^{\frac{-|j-i|}{Lp}}$, where $\mathbb{E}[.]$ is computed over all configurations of the chain induced by the dynamic randomness of the environment (equivalent to the thermal fluctuations in the environment in statistical physics).*

The first condition states that the expected magnitude of each bond vector is b_o. The second condition shows that the correlation between $|i - j|$ consecutive bond vectors is a decaying exponential with the correlation length Lp (persistence number), which is the number of time steps after which the bond vectors forget their original orientation. Note that despite the redundancy of the first condition, we choose to include it separately in order to emphasize the finite expected magnitude of the bond vectors.

3. Methods

We introduce the method of polymer-based exploration in reinforcement learning (PolyRL), which borrows concepts from Statistical Physics (de Gennes, 1979; Doi & Edwards, 1988) to induce persistent trajectories in continuous state-action spaces (Refer to Figure 1 for a schematic of simplified

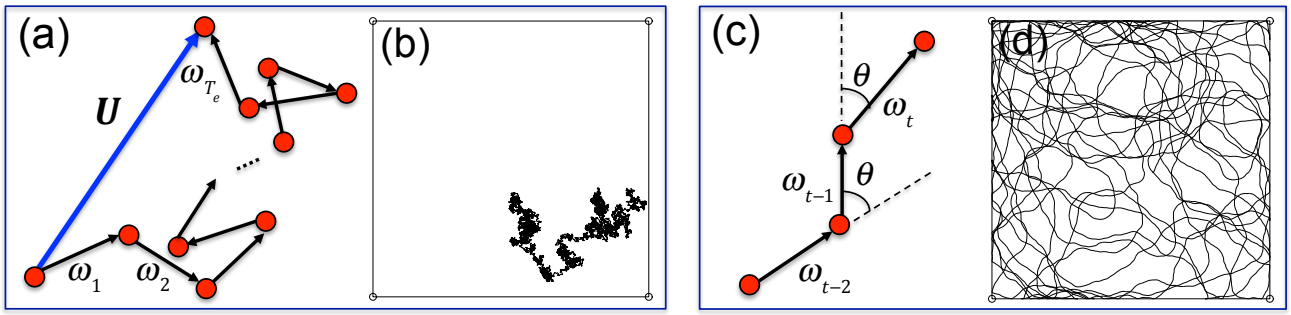

Figure 1. A chain (or trajectory) is shown as a sequence of T_e random bond vectors $\{\boldsymbol{\omega}_i\}_{i=1..T_e}$. In a freely-jointed chain (FJC) (a), the orientation of the bond vectors are independent of one another (Random Walk). In a freely-rotating chain (FRC) (c), the correlation angle θ is invariant between every two consecutive bond vectors, which induces a finite stiffness in the chain. (b, d) A qualitative comparison between an FJC (b) and an FRC with $\theta \approx 5.7^\circ$ (d), in a 2D environment of size 400×400 for 20000 number of steps.

polymer models and the Supplementary Information (Section 1) for more information regarding polymer models). As discussed below, our proposed technique balances exploration and exploitation using high-probability confidence bounds on a measure of spread in the state space. Algorithm 1 presents the PolyRL pseudo code. The method of action sampling is provided in Algorithm 2 in the Supplementary Information (Section 3).

The PolyRL agent chooses the sequence of actions in $\mathcal{A}$ such that every two consecutive action vectors are restricted in their orientation with the mean [correlation] angle θ. In order to induce persistent trajectories in the state space, the agent uses a measure of spread in the visited states in order to ensure the desired expansion of the trajectory $\tau_{\mathcal{S}}$ in $\mathcal{S}$. We define *radius of gyration squared*,

$$U_g^2(\tau_{\mathcal{S}}) := \frac{1}{T_e - 1} \sum_{s \in \tau_{\mathcal{S}}} d^2(s, \bar{\tau}_{\mathcal{S}}), \tag{4}$$

as a measure of the spread of the visited states in the state space $\mathcal{S}$, where $d(s, \bar{\tau}_{\mathcal{S}})$ is a metric defined on the state space $\mathcal{S}$, and serves as a measure of distance between the visited state s and the empirical mean of all visited states $\bar{\tau}_{\mathcal{S}}$. Also known as the center of mass, $\bar{\tau}_{\mathcal{S}}$ is calculated as, $\bar{\tau}_{\mathcal{S}} := \frac{1}{T_e} \sum_{s \in \tau_{\mathcal{S}}} s$.

At each time step, the agent calculates the radius of gyration squared (Equation 4) and compares it with the obtained value from the previous time step. If the induced trajectory in the state space is LSA-RW, it maintains an expected stiffness described by a bounded change in U_g^2. Refer to Theorems 3 and 4 in Section 4 for detailed information regarding the calculation of the bounds. In brief, the two theorems present high-probability confidence bounds on upper local sensitivity UB and lower local sensitivity LB, respectively. The lower bound ensures that the chain remains LSA-RW and expands in the space, while the upper bound prevents the agent from moving abruptly in the environment (The advantage of using LSA-RWs to explore the environment can be explained in terms of their high expansion rate, which is presented in Proposition 2 in the Supplementary Information (Section 1)). If the computed ΔU_g^2 is in the range $[LB, UB]$, the agent continues to perform PolyRL action sampling method (Algorithm 2 in the Supplementary Information). Otherwise, it samples the next action using the target policy. Due to the persistence of the PolyRL chain of exploratory actions, the orientation of the last greedy action is preserved for Lp (persistence) number of steps. As for the trajectories in the state space, upon observing correlation angles above $\pi/2$, the exploratory trajectory is broken and the next action is chosen with respect to the target policy π_μ.

4. Theory

In this section, we derive the upper and lower confidence bounds on the local sensitivity for the radius of gyration squared between $\tau_{\mathcal{S}}$ and $\tau'_{\mathcal{S}}$ (All proofs are provided in the Supplementary Information). Given the trajectory $\tau_{\mathcal{S}}$ and the corresponding radius of gyration squared $U_g^2(\tau_{\mathcal{S}})$ and persistence number $Lp_{\tau_{\mathcal{S}}} > 1$, we seek a description for the permissible range of $U_g^2(\tau'_{\mathcal{S}})$ such that the stiffness of the trajectory is preserved. Note that the derived equations for the upper and lower confidence bounds are employed in the PolyRL algorithm (Algorithm 1) in bounding the change in U_g^2 to ensure the formation of locally self-avoiding walks in the trajectory of visited states in the state space.

High-probability upper bound - We define the upper local sensitivity on U_g^2 upon observing new state $s_{T_e} \in \mathcal{S}$ as,

$$ULS_{Ug^2}(\tau_{\mathcal{S}}) := \sup_{s_{T_e} \in \Omega} U_g^2(\tau'_{\mathcal{S}}) - U_g^2(\tau_{\mathcal{S}}). \tag{5}$$

Given the observed state trajectory $\tau_{\mathcal{S}}$ with persistence number $Lp_{\tau_{\mathcal{S}}}$, the upper local sensitivity ULS_{Ug^2} provides the maximum change observed upon visiting the next accessible

Algorithm 1 PolyRL Algorithm

Require: Exploration factor β, Average correlation angle θ and Correlation variance σ^2
for N in total number of episodes **do**
$\delta \leftarrow 1 - e^{-\beta N}$ {An increasing function of the the episode number.}
Sample $\mathbf{a}_0$ and $\mathbf{s}_0$ from an initial state-action distribution, and Exploit-flag $\leftarrow 0$ {If Exploit-flag is 1, the agent uses the target policy to select action, and explores otherwise.}
repeat
if Exploit-flag == 1 **then**
Draw a random number $\kappa \sim \mathcal{N}(0,1)$
if $\kappa \leq \delta$ **then**
$\mathbf{a}_t \sim \pi_\mu$ {action is drawn from the target policy}
else
Exploit-flag $\leftarrow 0$
Start a new exploratory trajectory by setting the radius of gyration squared to zero
Draw random number $\eta \sim \mathcal{N}(\theta, \sigma^2)$
$\mathbf{a}_t \sim \pi_{\text{PolyRL}}(\eta, \mathbf{a_{t-1}})$
end if
else
Compute the change in the radius of gyration squared ΔU_g^2 letting $d = L_2$-norm and using eq. (4).
Calculate UB and LB using eqs. (7) and (11).
if $\Delta U_g^2 \geq LB$ and $\Delta U_g^2 \leq UB$ **then**
Draw random number $\eta \sim \mathcal{N}(\theta, \sigma^2)$
$\mathbf{a}_t \sim \pi_{\text{PolyRL}}(\eta, \mathbf{a_{t-1}})$
else
Exploit-flag $\leftarrow 1$, and $\mathbf{a}_t \sim \pi_\mu$
end if
end if
until the agent reaches an absorbing state or the maximum allowed time steps in an episode
end for

state $s' \in \Omega$. With the goal of constructing the new trajectory $\tau'_{\mathcal{S}}$ such that it preserves the stiffness induced by $\tau_{\mathcal{S}}$, we calculate the high-probability upper confidence bound on ULS_{Ug^2}. To do so, we write the term $d^2(s, \bar{\tau}_{\mathcal{S}})$ in Equation 4 as a function of bond vectors ω_i, which is presented in Lemma 2 given that $d = L_2$-norm. We further substitute the resulting $U_g^2(\tau_{\mathcal{S}})$ in Equation 5 with the obtained expression from Equation 4.

Lemma 2 *Let $\tau_{\mathcal{S}} = (s_0, \ldots, s_{T_e-1})$ be the trajectory of visited states, s_{T_e} be a newly visited state and $\boldsymbol{\omega}_i = s_i - s_{i-1}$ be the bond vector that connects two consecutive visited states s_{i-1} and s_i. Then we have,*

$$\|s_{T_e} - \bar{\tau}_{\mathcal{S}}\|^2 = \|\boldsymbol{\omega}_{T_e} + \frac{1}{T_e}\left[\sum_{i=1}^{T_e-1} i\boldsymbol{\omega}_i\right]\|^2. \tag{6}$$

The result of Lemma 2 will be used in the proof of Theorem 3, as shown in the Supplementary Information (Section 2). In Theorem 3, we provide a high-probability upper bound on $ULS_{Ug^2}(\tau_{\mathcal{S}})$.

Theorem 3 (Upper-Bound Theorem) *Let $\delta \in (0,1)$ $\tau_{\mathcal{S}}$ be an LSA-RW in $\mathcal{S}$ induced by PolyRL with the persistence number $Lp_{\tau_{\mathcal{S}}} > 1$ within episode N, $\omega_{\tau_{\mathcal{S}}} = \{\boldsymbol{\omega}_i\}_{i=1}^{T_e-1}$ be the sequence of corresponding bond vectors, where $T_e > 0$ denotes the number of bond vectors within $\tau_{\mathcal{S}}$, and b_o be the average bond length. The upper confidence bound for $ULS_{Ug^2}(\tau_{\mathcal{S}})$ with probability of at least $1-\delta$ is,*

$$\begin{aligned} UB =& \Lambda(T_e, \tau_{\mathcal{S}}) \\ &+ \frac{1}{\delta}\left[\Gamma(T_e, b_o, \tau_{\mathcal{S}}) + \frac{2b_o^2}{T_e^2}\sum_{i=1}^{T_e-1} i e^{\frac{-(T_e-i)}{Lp_{\tau_{\mathcal{S}}}}}\right], \end{aligned} \tag{7}$$

where,

$$\Lambda(T_e, \tau_{\mathcal{S}}) = -\frac{1}{T_e - 1}U_g^2(\tau_{\mathcal{S}}) \tag{8}$$

$$\Gamma(T_e, b_o, \tau_{\mathcal{S}}) = \frac{b_o^2}{T_e} + \frac{\|\sum_{i=1}^{T_e-1} i\boldsymbol{\omega}_i\|^2}{T_e^3} \tag{9}$$

Equation 7 provides an upper bound on the pace of the trajectory expansion in the state space (denoted by $\Delta U_g^2 \leq UB$ in Algorithm 1) to prevent the agent from moving abruptly in the environment, which would otherwise lead to a break in its temporal correlation with the preceding states. Similarly, we introduce the lower local sensitivity LLS_{Ug^2}, which provides the minimum change observed upon visiting the next accessible state $s' \in \Omega$.

High-probability lower bound - In this part, We define the lower local sensitivity on U_g^2 upon observing new state $s_{T_e} \in \mathcal{S}$ as,

$$LLS_{Ug^2}(\tau_{\mathcal{S}}) := \inf_{s_{T_e} \in \Omega} U_g^2(\tau'_{\mathcal{S}}) - U_g^2(\tau_{\mathcal{S}}). \tag{10}$$

We further compute the high-probability lower confidence bound on LLS_{Ug^2} in order to guarantee the expansion of the trajectory $\tau_{\mathcal{S}}$ upon visiting the next state.

Theorem 4 (Lower-Bound Theorem) *Let $\delta \in (0,1)$ and $\tau_{\mathcal{S}}$ be an LSA-RW in $\mathcal{S}$ induced by PolyRL with the persistence number $Lp_{\tau_{\mathcal{S}}} > 1$ within episode N, $\omega_{\tau_{\mathcal{S}}} = \{\boldsymbol{\omega}_i\}_{i=1}^{T_e-1}$ be the sequence of corresponding bond vectors, where $T_e > 0$ denotes the number of bond vectors within $\tau_{\mathcal{S}}$,*

and b_o be the average bond length. The lower confidence bound on $LLS_{Ug^2}(\tau_{\mathcal{S}})$ at least with probability $1-\delta$ is,

$$LB = \Lambda(T_e, \tau_{\mathcal{S}}) + (1 - \sqrt{2 - 2\delta}) \times \left[\Gamma(T_e, b_o, \tau_{\mathcal{S}}) + \frac{(T_e - 1)(T_e - 2)}{T_e^2} b_0^2 e^{\frac{-|T_e - 1|}{L_p \tau_{\mathcal{S}}}} \right], \quad (11)$$

where,

$$\Lambda(T_e, \tau_{\mathcal{S}}) = -\frac{1}{T_e - 1} U_g^2(\tau_{\mathcal{S}}) \quad (12)$$

$$\Gamma(T_e, b_o, \tau_{\mathcal{S}}) = \frac{b_o^2}{T_e} + \frac{\| \sum_{i=1}^{T_e - 1} i \boldsymbol{\omega}_i \|^2}{T_e^3} \quad (13)$$

Equation 11 provides a lower bound on the change in the expansion of the trajectory (*e.g.* $\Delta U_g^2 \geq LB$ as shown in Algorithm 1).

The factor $\delta \in [0, 1]$ that arises in Equations 7 and 11 controls the tightness of the confidence interval. In order to balance the trade-off between exploration and exploitation, we choose to increase δ with time, as increasing δ leads to tighter bounds and thus, higher probability of exploitation. In addition, the exploration factor δ determines the probability of switching from exploitation back to starting a new exploratory trajectory. Note that for experimental purposes, we let $\delta = 1 - e^{-\beta N}$, where the exploration factor $\beta \in (0, 1)$ is a hyper parameter and N is the number of elapsed episodes.

The following corollary is an immediate consequence of Theorems 3 and 4 together with Assumption 1.

Corollary 5 *Given that Assumption 1 is satisfied, any exploratory trajectory induced by PolyRL Algorithm (ref. Algorithm 1) with high probability is an LSA-RW.*

The proof is provided and discussed in the Supplementary Information (Section 2).

5. Related Work

A wide range of exploration techniques with theoretical guarantees (*e.g.* PAC bounds) have been developed for MDPs with finite or infinitely countable state or action spaces (Kearns & Singh, 2002; Brafman & Tennenholtz, 2002; Strehl & Littman, 2004; Lopes et al., 2012; Azar et al., 2017; White & White, 2010; Wang et al., 2020), however extending these algorithms to real-world settings with continuous state and action spaces without any assumption on the structure of state-action spaces or the reward function is impractical (Haarnoja et al., 2017; Houthooft et al., 2016a).

Perturbation-based exploration strategies are, by nature, agnostic to the structure of the underlying state-action spaces and are thus suitable for continuous domains. Classic perturbation-based exploration strategies typically involve a perturbation mechanism at either the action-space or the policy parameter-space level. These methods subsequently employ stochasticity at the policy level as the main driving force for exploration (Deisenroth et al., 2013). Methods that apply perturbation at the parameter level often preserve the noise distribution throughout the trajectory (Ng & Jordan, 2000; Sehnke et al., 2010; Theodorou et al., 2010; Fortunato et al., 2017; Colas et al., 2018; Plappert et al., 2018), and do not utilize the trajectory information in this regard. Furthermore, a majority of action-space perturbation approaches employ per-step independent or correlated perturbation (Wawrzyński, 2009; Lillicrap et al., 2015; Haarnoja et al., 2017; Xu et al., 2018). For instance, (Lillicrap et al., 2015) uses the Ornstein–Uhlenbeck (OU) process to produce auto-correlated additive noise at the action level and thus benefit from the correlated noise between consecutive actions.

While maintaining correlation between consecutive actions is advantageous in many locomotion tasks (Morimoto & Doya, 2001; Kober et al., 2013; Gupta et al., 2018), it brings technical challenges due to the non-Markovian nature of the induced decision process (Perez & Silander, 2017), which leads to substantial dependence on the history of the visited states. This challenge is partially resolved in the methods that benefit from some form of parametric memory (Plappert et al., 2018; Lillicrap et al., 2015; Sharma et al., 2017). However, they all suffer from the lack of *informed orientational persistence*, *i.e.* the agent's persistence in selecting actions that preserve the orientation of the state trajectory induced by the target policy. Particularly in sparse-reward structured environments, where the agent rarely receives informative signals, the agent will eventually get stuck in a small region since the analytical change on the gradient of greedy policy is minimal (Hare, 2019; Kang et al., 2018). Hence, the agent's persistent behaviour in action sampling with respect to the local history (short-term memory) of the selected state-action pairs plays a prominent role in exploring the environments with sparse or delayed reward structures.

6. Experiments

We assess the performance of PolyRL in comparison with that of other state-of-the-art exploration approaches combined with off-policy learning methods. In particular, we integrate PolyRL with three learning algorithms: (1) the Q-learning method ((Watkins & Dayan, 1992)) with linear function approximation in a 2D sparse continuous state-and-action navigation task, where the performance of PolyRL is compared with that of ϵ-greedy; (2) the Soft Actor-Critic (SAC) algorithm (Haarnoja et al., 2018), where PolyRL is combined with SAC (SAC-PolyRL) and replaces the random exploration phase during the first $10,000$ steps in the

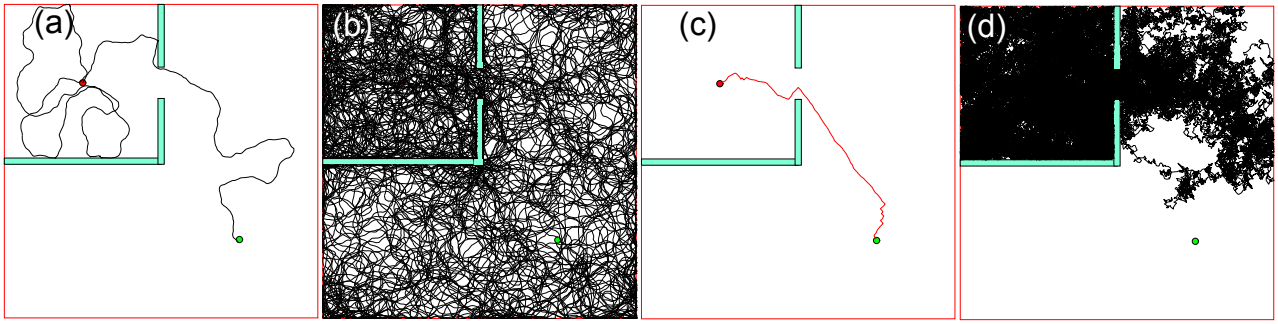

Figure 2. Visualization of the agent's trajectory in a 2D navigation task with continuous state-action space and sparse-reward structure. The environment is composed of a 100×100 big chamber and a 50×50 smaller chamber. The start state is located inside the small chamber (the red circle with a diameter of 1), and the goal is outside the small chamber (the green circle with a diameter of 1). The agent receives $+100$ reward when it reaches the goal and 0 reward elsewhere. (a-c) Performance of the PolyRL agent with the correlation angle $\theta \simeq 0.2$. (a) A sample trajectory of the PolyRL agent in one episode. (b) Coverage of the environment after 11 episodes. (c) A sample trajectory of the PolyRL agent during the evaluation phase after learning the task. (d) The ϵ-greedy agent's coverage of the environment after 11 episodes.

SAC algorithm and is subsequently compared with SAC as well as Optimistic Actor-Critic (OAC) (Ciosek et al., 2019) methods; and (3) the deep deterministic policy gradients (DDPG) (Lillicrap et al., 2015) algorithm, where PolyRL (DDPG-PolyRL) is assessed in comparison with additive uncorrelated Gaussian action space noise (DDPG-UC), correlated Ornstein-Uhlenbeck action space noise (DDPG-OU) (Uhlenbeck & Ornstein, 1930; Lillicrap et al., 2015), adaptive parameter space noise (DDPG-PARAM) (Plappert et al., 2017), as well as the Fine Grained Action Repetition (DDPG-FiGAR) (Sharma et al., 2017) method.

The sets of experiments involving the learning methods DDPG and SAC are performed in MuJoCo high-dimensional continuous control tasks "SparseHopper-V2" ($\mathcal{A} \subset \mathbb{R}^3$, $\mathcal{S} \subset \mathbb{R}^{11}$), "SparseHalfCheetah-V2" ($\mathcal{A} \subset \mathbb{R}^6$, $\mathcal{S} \subset \mathbb{R}^{17}$), and "SparseAnt-V2" ($\mathcal{A} \subset \mathbb{R}^8$, $\mathcal{S} \subset \mathbb{R}^{111}$) (Refer to the Supplementary Information for the benchmark results of the same algorithms in the standard (dense-reward) MuJoCo tasks).

Algorithm and Environment Settings - The environment in our 2D sparse-reward navigation tasks either consists of only one 400×400 chamber (goal reward $+1000$), or a 50×50 room encapsulated by a 100×100 chamber. Initially positioned inside the small room, the agent's goal in the latter case is to find its way towards the bigger chamber, where the goal is located (goal reward $+100$) (Figure 2). Moreover, in order to make the former task more challenging, in a few experiments, we introduce a *puddle* in the environment, where upon visiting, the agent receives the reward -100. In order to assess the agent's performance, we integrate the PolyRL exploration algorithm with the Q-learning method with linear function approximation (learning rate $= 0.01$) and compare the obtained results with those of the ϵ-greedy exploration with Q-learning. We subsequently plot the quantitative results for the former task and visualize the resulting trajectories for the latter.

In the sparse MuJoCo tasks, the agent receives a reward of $+1$ only when it crosses a target distance λ, termed the *sparsity threshold*. Different λ values can change the level of difficulty of the tasks significantly. Note that due to the higher performance of SAC-based methods compared with that of DDPG-based ones, we have elevated λ for SAC-based experiments, making the tasks more challenging. Moreover, we perform an exhaustive grid search over the corresponding hyper parameters for each task. The sparsity thresholds, the obtained hyper-parameter values, as well as the network architecture of the learning algorithms DDPG and SAC are provided in the Supplementary Information (Section 5).

Results and Discussion - We present the qualitative results for the 2D sparse navigation task in Figure 2. An example of PolyRL trajectory in one episode (Figure 2 (a)) demonstrates the expansion of the PolyRL agent trajectory in the environment. After 11 episodes, the PolyRL agent exhibits a full coverage of the environment (Figure 2 (b)) and is consequently able to learn the task (Figure 2 (c)), while the ϵ-greedy agent is not able to reach the goal state even once, and thus fails to learn the task (Figure 2 (d)). This visual observation highlights the importance of space coverage in sparse-reward tasks, where the agent rarely receives informative reinforcement from the environment. An effective trajectory expansion in the environment exposes the agent to the unvisited regions of the space, which consequently increases the frequency of receiving informative reinforcement and accelerates the learning process. In Figure 3, the quantitative results for learning the task in a similar environ-

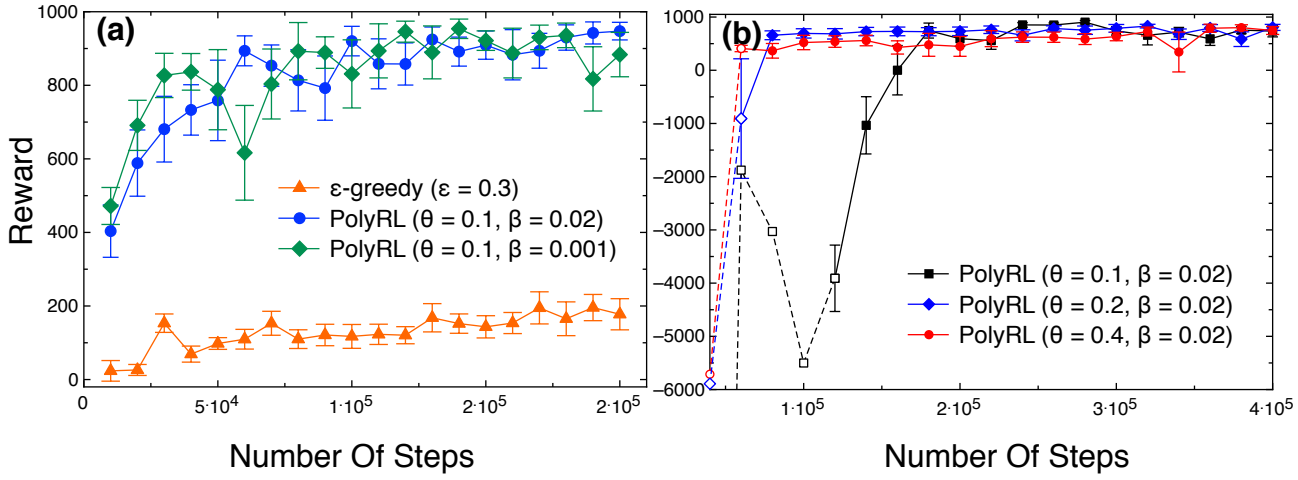

Figure 3. The performance of PolyRL and ϵ-greedy in a 2D continuous sparse-reward navigation task (the 400×400 chamber). (a) The environment does not include a puddle. (b) The environment contains a puddle. The first point on the $\theta = 0.1$ curve is not shown on the graph due to its minimal average reward value ($\simeq -30000$), which affects the range of the y-axis. It is connected to the next point via a dashed line. The data points on the dashed lines do not include error bars because of substantial variances. For the same reason (out of range values and large variances), the results for the ϵ-greedy exploration method are not shown here.

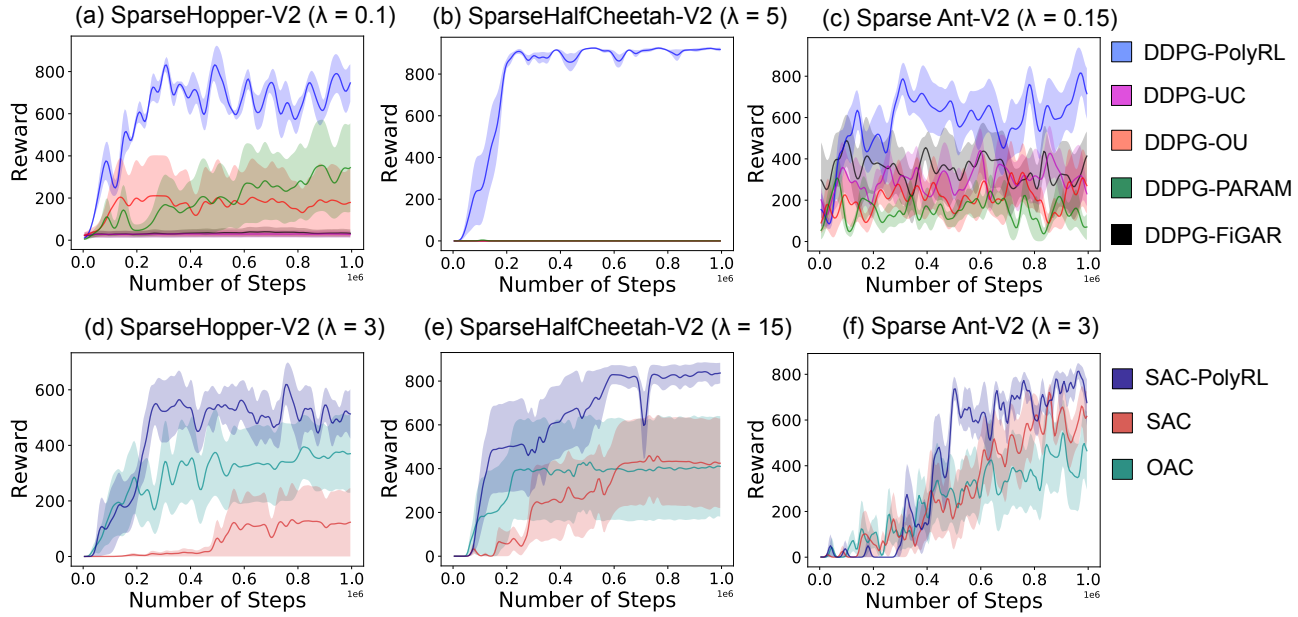

Figure 4. The simulation results on "SparseHopper-v2" (a and d), "SparseHalfCheetah-v2" (b and e) and "SparseAnt-v2" (c and f). The results are averaged over 5 random seeds. The error bars depict the standard error on the mean (Refer to the Supplementary Information for the benchmark results obtained from the baseline algorithms).

ment (the 400×400 chamber) are shown in both the absence (Figure 3 (a)) and presence (Figure 3 (b)) of a puddle. In both cases, the PolyRL exploration method outperforms ϵ-greedy. In Figure 3 (b), we observe that trajectories with lower persistence (larger θ) present a better performance compared with stiffer trajectories. Note that due to the existence of walls in these 2D tasks, the transition probability kernel is specifically non-smooth at the walls. Thus, the smoothness assumption on the transition probability kernel made earlier for the theoretical convenience does not apply in these specific environments. Yet, we empirically show that PolyRL still achieves a high performance in learning

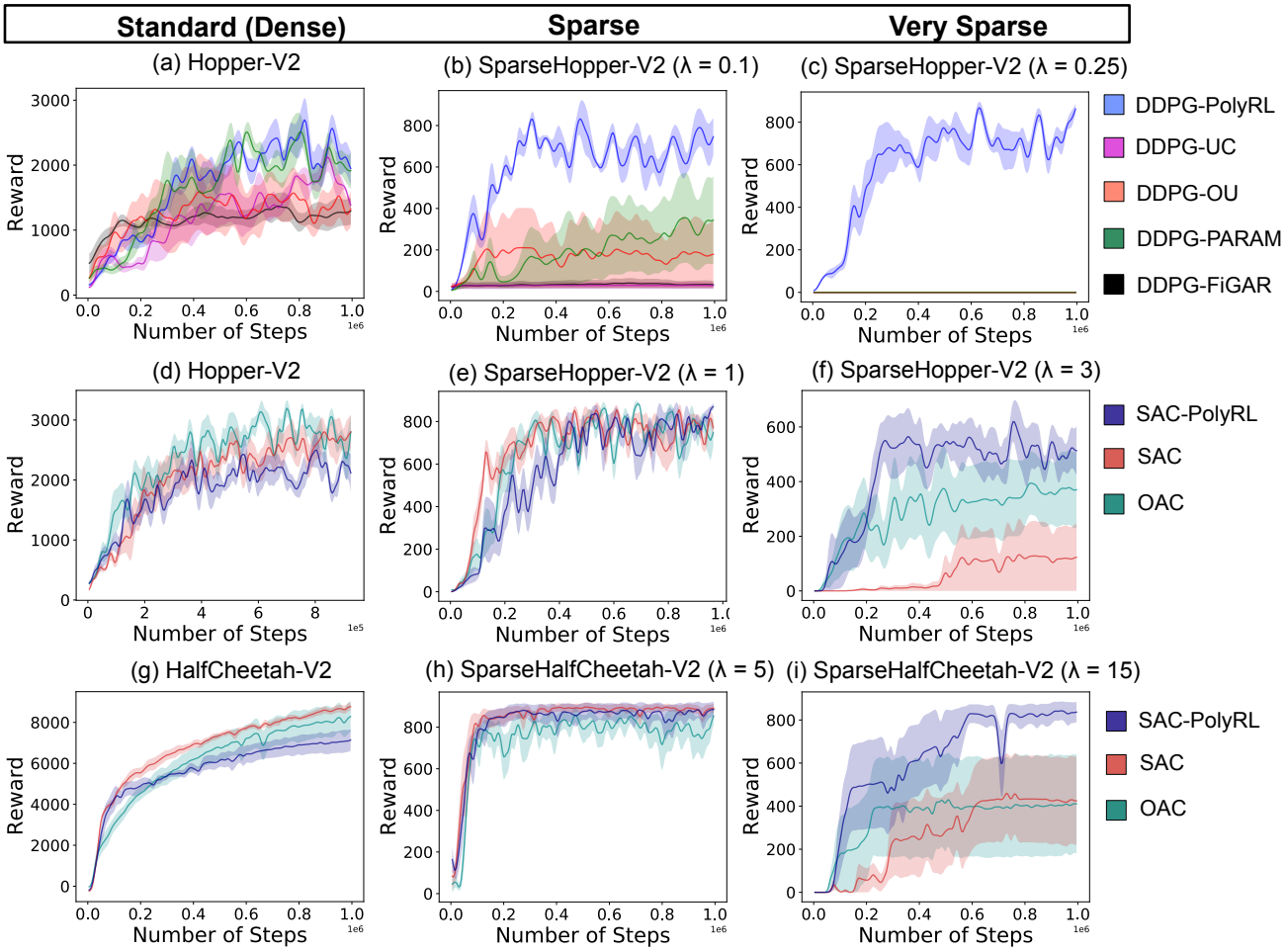

Figure 5. PolyRL sensitivity to the change of sparsity threshold in some sample experiments. The column on the left depicts the results for standard (with dense reward) MuJoCo tasks. From left to right, the sparsity of the reward structure increases.

these tasks.

We illustrate the plotted results for the sparse MuJoCo tasks in Figure 4. The obtained results for DDPG-based and SAC-based algorithms in SparseHopper-V2 (Figures 4 (a, d)), SparseHalfCheetah-V2 (Figures 4 (b, e)), and SparseAnt-V2 (Figures 4 (c, f)) show that integrating the two learning methods with the exploration algorithm PolyRL leads to improvement in their performance. The results achieved by PolyRL exploration method confirms that the agent has been able to efficiently and sufficiently cover the space, receive informative reinforcement, and learn the tasks. The high performance of SAC-PolyRL is particularly significant, in the sense that PolyRL assists SAC in the data generation process only for the first $10,000$ steps. Yet, this short presence leads to a notable contribution in enhancing the performance of SAC.

Another notable feature of PolyRL is its relatively low sensitivity to the increase in sparsity threshold λ compared to that of DDPG-based and SAC-based algorithms. Figure 5 illustrates the performance of PolyRL in some sample experiments for three different sparsity thresholds. The level of complexity of the tasks increases from left to right with the sparsity threshold λ. As the sparsity level changes from sparse to very sparse, the graphs demonstrate a sharp decrease in the performance of PolyRL counterparts, while the PolyRL performance stays relatively stable (Note that due to the reward structure in these sparse tasks and considering that the maximum number of steps in each episode is by default set to 1000 in the Gym environments, the maximum reward that an agent could get during each evaluation round cannot exceed 1000). This PolyRL agent's behaviour can be explained by its relatively fast expansion in the space (Refer to Proposition 2 in the Supplementary Information (Section 1)), which leads to faster access to the sparsely distributed rewards compared with other DDPG-based and SAC-based methods. On the other hand, in standard tasks, where the reinforcement is accessible to the agents at each

time-step, as PolyRL does not use the received rewards in its action selection process, it might skip the informative signals nearby and move on to the farther regions in the space, leading to possibly acquiring less amount of information and lower performance. In other words, the strength of PolyRL is most observable in the tasks where accessing information is limited or delayed.

7. Conclusion

We propose a new exploration method in reinforcement learning (PolyRL), which leverages the notion of locally self-avoiding random walks and is designed for environments with continuous state-action spaces and sparse-reward structures. The most interesting aspect of our proposal is the explicit construction of each exploratory move based on the entire existing trajectory, rather than just the current observed state. While the agent chooses its next move based on its current state, the inherent locally self-avoiding property of the walk acts as an implicit memory, which governs the agent's exploratory behaviour. Yet this locally controlled behavior leads to an interesting global property for the trajectory, which is an improvement in the coverage of the environment. This feature, as well as not relying on extrinsic rewards in the decision-making process, makes PolyRL perfect for the sparse-reward tasks. We assess the performance of PolyRL in 2D continuous sparse navigation tasks, as well as three sparse high-dimensional simulation tasks, and show that PolyRL performs significantly better than the other exploration methods in combination with the baseline learning algorithms DDPG and SAC. Moreover, we show that compared to the performance of PolyRL counterparts, the performance of our proposed method has lower sensitivity to the increase in the sparsity of the extrinsic reward, and thus its performance stays relatively stable. Finally, a more adaptive version of PolyRL, which can map the changes in the action trajectory stiffness to that of the state trajectory, could be helpful in more efficient learning of the simulation tasks.

Acknowledgements

The authors would like to thank Prof. Walter Reisner for providing valuable feedback on the initial direction of this work, and Riashat Islam for helping with the experiments in the early stages of the project. Computing resources were provided by Compute Canada and Calcul Québec throughout the project, and by Deeplite Company for preliminary data acquisition, which the authors appreciate. Funding is provided by Natural Sciences and Engineering Research Council of Canada (NSERC).

References

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Pieter Abbeel, O., and Zaremba, W. Hindsight experience replay. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 5048–5058. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7090-hindsight-experience-replay.pdf.

Antos, A., Szepesvári, C., and Munos, R. Fitted q-iteration in continuous action-space mdps. In *Advances in neural information processing systems*, pp. 9–16, 2008.

Azar, M. G., Osband, I., and Munos, R. Minimax regret bounds for reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 263–272. JMLR. org, 2017.

Bartlett, P. L. and Tewari, A. Sample complexity of policy search with known dynamics. In *Advances in Neural Information Processing Systems*, pp. 97–104, 2007.

Brafman, R. I. and Tennenholtz, M. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct): 213–231, 2002.

Ciosek, K., Vuong, Q., Loftin, R., and Hofmann, K. Better exploration with optimistic actor-critic. *arXiv preprint arXiv:1910.12807*, 2019.

Colas, C., Sigaud, O., and Oudeyer, P.-Y. GEP-PG: Decoupling exploration and exploitation in deep reinforcement learning algorithms. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1039–1048, 2018.

Dabney, W., Ostrovski, G., and Barreto, A. Temporally-extended, epsilon-greedy exploration. *arXiv preprint arXiv:2006.01782*, 2020.

de Gennes, P.-G. *Scaling concepts in polymer physics*. Cornel University Press, 1979.

Deisenroth, M. P., Neumann, G., Peters, J., et al. A survey on policy search for robotics. *Foundations and Trends® in Robotics*, 2(1–2):1–142, 2013.

Doi, M. and Edwards, S. F. *The theory of polymer dynamics*, volume 73. oxford university press, 1988.

Fortunato, M., Azar, M. G., Piot, B., Menick, J., Osband, I., Graves, A., Mnih, V., Munos, R., Hassabis, D., Pietquin, O., et al. Noisy networks for exploration. *arXiv preprint arXiv:1706.10295*, 2017.

Fu, J., Co-Reyes, J., and Levine, S. Ex2: Exploration with exemplar models for deep reinforcement learning. In *Advances in neural information processing systems*, pp. 2577–2587, 2017.

Gupta, A., Mendonca, R., Liu, Y., Abbeel, P., and Levine, S. Meta-reinforcement learning of structured exploration strategies. In *Advances in Neural Information Processing Systems*, pp. 5302–5311, 2018.

Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. Reinforcement learning with deep energy-based policies. In *Proceedings of the International Conference on Machine Learning*, 2017.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pp. 1861–1870, 2018.

Hare, J. Dealing with sparse rewards in reinforcement learning. *arXiv preprint arXiv:1910.09281*, 2019.

Houthooft, R., Chen, X., Chen, X., Duan, Y., Schulman, J., Turck, F. D., and Abbeel, P. VIME: variational information maximizing exploration. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 1109–1117, 2016a.

Houthooft, R., Chen, X., Duan, Y., Schulman, J., De Turck, F., and Abbeel, P. Vime: Variational information maximizing exploration. In *Advances in Neural Information Processing Systems*, pp. 1109–1117, 2016b.

Kang, B., Jie, Z., and Feng, J. Policy optimization with demonstrations. In *International Conference on Machine Learning*, pp. 2469–2478, 2018.

Kearns, M. and Singh, S. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2-3):209–232, 2002.

Kober, J., Bagnell, J. A., and Peters, J. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

Lakshminarayanan, A. S., Sharma, S., and Ravindran, B. Dynamic action repetition for deep reinforcement learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pp. 2133–2139, 2017.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Lopes, M., Lang, T., Toussaint, M., and Oudeyer, P.-Y. Exploration in model-based reinforcement learning by empirically estimating learning progress. In *Advances in Neural Information Processing Systems*, pp. 206–214, 2012.

Morimoto, J. and Doya, K. Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning. *Robotics and Autonomous Systems*, 36(1):37–51, 2001.

Nair, A., McGrew, B., Andrychowicz, M., Zaremba, W., and Abbeel, P. Overcoming exploration in reinforcement learning with demonstrations. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6292–6299. IEEE, 2018.

Ng, A. Y. and Jordan, M. Pegasus: a policy search method for large mdps and pomdps. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pp. 406–415, 2000.

Ostrovski, G., Bellemare, M. G., van den Oord, A., and Munos, R. Count-based exploration with neural density models. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2721–2730. JMLR. org, 2017.

Perez, J. and Silander, T. Non-markovian control with gated end-to-end memory policy networks. *arXiv preprint arXiv:1705.10993*, 2017.

Plappert, M., Houthooft, R., Dhariwal, P., Sidor, S., Chen, R. Y., Chen, X., and othere. Parameter space noise for exploration. *arXiv preprint arXiv:1706.01905*, 2017.

Plappert, M., Houthooft, R., Dhariwal, P., Sidor, S., Chen, R. Y., Chen, X., Asfour, T., Abbeel, P., and Andrychowicz, M. Parameter space noise for exploration. In *International Conference on Learning Representations*, 2018.

Sehnke, F., Osendorfer, C., Rückstieß, T., Graves, A., Peters, J., and Schmidhuber, J. Parameter-exploring policy gradients. *Neural Networks*, 23(4):551–559, 2010.

Sharma, S., Lakshminarayanan, A. S., and Ravindran, B. Learning to repeat: Fine grained action repetition for deep reinforcement learning. *arXiv preprint arXiv:1702.06054*, 2017.

Strehl, A. and Littman, M. Exploration via model based interval estimation. In *International Conference on Machine Learning*. Citeseer, 2004.

Taïga, A. A., Fedus, W., Machado, M. C., Courville, A., and Bellemare, M. G. Benchmarking bonus-based exploration methods on the arcade learning environment. *arXiv preprint arXiv:1908.02388*, 2019.

Theodorou, E., Buchli, J., and Schaal, S. A generalized path integral control approach to reinforcement learning. *Journal of Machine Learning Research*, 11(Nov):3137–3181, 2010.

Uhlenbeck, G. E. and Ornstein, L. S. On the theory of the brownian motion. *Physical review*, 36(5):823, 1930.

Wang, Y., Dong, K., Chen, X., and Wang, L. Q-learning with ucb exploration is sample efficient for infinite-horizon mdp. In *International Conference on Learning Representations*, 2020.

Watkins, C. J. and Dayan, P. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Wawrzyński, P. Real-time reinforcement learning by sequential actor–critics and experience replay. *Neural Networks*, 22(10):1484–1497, 2009.

White, M. and White, A. Interval estimation for reinforcement-learning algorithms in continuous-state domains. In *Advances in Neural Information Processing Systems*, pp. 2433–2441, 2010.

Xu, T., Liu, Q., Zhao, L., and Peng, J. Learning to explore with meta-policy gradient. In *Proceedings of the International Conference on Machine Learning*, volume 80, pp. 5459–5468, 2018.