Latent Space Exploration and Trajectory Space Update in Temporally-Correlated Episodic Reinforcement Learning

Ge Li[1], Hongyi Zhou[1], Dominik Roth[1], Serge Thilges[1], Fabian Otto[2], Rudolf Lioutikov[1], Gerhard Neumann[1]

***Abstract*— Current advancements in reinforcement learning (RL) have predominantly focused on learning step-based policies that generate actions for each perceived state. While these methods efficiently leverage step information from environmental interaction, they often ignore the temporal correlation between actions, resulting in inefficient exploration and unsmooth trajectories that are challenging to implement on real hardware. Episodic RL (ERL) seeks to overcome these challenges by exploring in parameters space that capture the correlation of actions. However, these approaches typically compromise data efficiency, as they treat trajectories as opaque *black boxes*. In this work, we introduce a novel ERL algorithm, Temporally-Correlated Episodic RL (TCE), which effectively utilizes step information in episodic policy updates, opening the 'black box' in existing ERL methods while retaining the smooth and consistent exploration in parameter space. TCE synergistically combines the advantages of step-based and episodic RL, achieving comparable performance to recent ERL methods while maintaining data efficiency akin to state-of-the-art (SoTA) step-based RL.**

I. Introduction

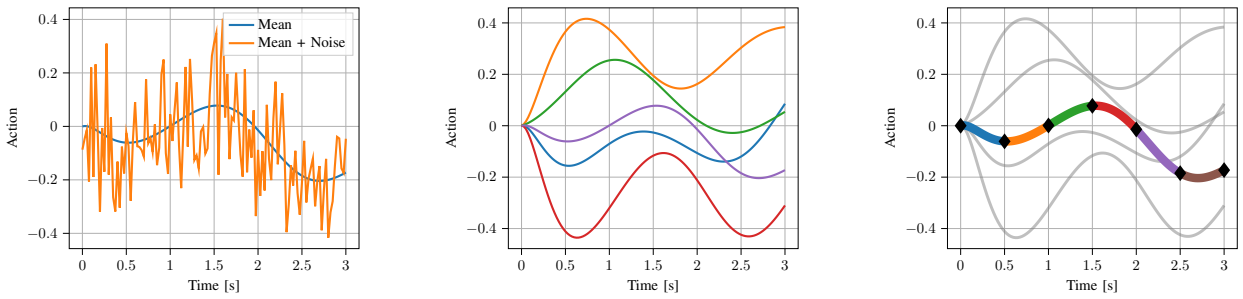
By considering how policies interact with the environment, reinforcement learning (RL) methodologies can be classified into two distinct categories: step-based RL (SRL) and episodic RL (ERL). SRL predicts actions for each perceived state, while ERL selects an entire behavioral sequence at the start of an episode. Most predominant deep RL methods, such as PPO [1] and SAC [2], fall into the category of SRL. In these methods, the step information — comprising state, action, reward, subsequent state, and done signal received by the RL agent at each discrete time step — is pivotal for policy updates. This granular data aids in estimating the policy gradient [3, 4], approximating state or state-action value functions [2], and assessing advantages [5]. Although SRL methods have achieved great success in various domains, they often face significant exploration challenges. Exploration in SRL, often based on a stochastic policy like a factorized Gaussian, typically lacks temporal and cross-DoF (degrees of freedom) correlations. This deficiency leads to inconsistent and inefficient exploration across state and action spaces [6, 7], as shown in Figure 1a. Furthermore, the high variance in trajectories generated through such exploration can cause suboptimal convergence and training instability, a phenomenon highlighted by considerable performance differences across various random seeds [8].

Episodic RL, in contrast to SRL, represents a distinct branch of RL that emphasizes the maximization of returns over entire episodes [9–11], rather than focusing on the internal evolution of the environment interaction. This approach shifts the solution search from per-step actions to a parameterized trajectory space, employing techniques like Movement Primitives (MPs) [12, 13]. Such exploration strategy allows for broader exploration horizons and ensures consistent trajectory smoothness across task episodes, as illustrated in Figure 1b. Additionally, it is theoretically capable of capturing temporal correlations and interdependencies among DoF. ERL typically treats entire trajectories as single data points, often overlooking the internal changes in the environment and state transitions. This approach leads to training predominantly using black-box optimization methods [14–17]. The term *black box* in our title reflects this reliance on black-box optimization, which tends to overlook detailed step-based information acquired during environmental interactions. However, this often results in a lack of attention to the individual contributions of each segment of the trajectory to the overall task success. Consequently, while ERL excels in expansive exploration and maintaining trajectory smoothness, it typically requires a larger volume of samples for effective policy training. In contrast, step-based RL methods have demonstrated notable advancements in learning efficiency by utilizing this detailed step-based information.

Open the Black Box. In this paper, our goal is to integrate step-based information into the policy update process of ERL. Our proposed method, Temporally-Correlated Episodic RL (TCE), moves beyond the traditional approach of treating an entire trajectory as a single data point. Instead, we transform trajectory-wide elements, such as reproducing likelihood and advantage, into their segment-wise counterparts. This enables us to leverage the step-based information to recognize and accentuate the unique contributions of each trajectory segment to overall task success. Through this innovative approach, we have opened the black box of ERL, making it more effective while retaining its strength. As a further step, we explore the benefits of fully-correlated trajectory exploration in deep ERL. We demonstrate that leveraging full covariance matrices for trajectory distributions significantly improves policy quality in existing black-box ERL methods like [17].

Our contributions are summarized as: (a) We propose TCE, a novel RL framework that integrates step-based information into the policy updates of ERL, while preserving the broad exploration scope and trajectory smoothness characteristic of ERL. (b) We provide an in-depth analysis of exploration strategies that effectively capture both temporal and degrees of freedom (DoF) correlations, demonstrating their

[1]Karlsruhe Institute of Technology, Germany. ge.li@kit.edu
[2]Microsoft Research, UK.



(a) Step-based RL (b) Traj.-based, Episodic RL (c) TCE (ours)

Fig. 1: Illustration of exploration strategies: (a) SRL samples actions by adding noise to the predicted mean, resulting in inconsistent exploration and jerky actions. However, their leverage of step-based information leads to efficient policy updates. (b) ERL samples complete trajectories in a parameter space and generate consistent control signals. Yet, they often treat trajectories as single data points and overlook the step-based information during the interaction, causing inefficient policy update. (c) TCE combines the benefits of both, using per-step information for policy update while sampling complete trajectories with broader exploration and high smoothness.

beneficial impact on policy quality and trajectory smoothness. (c) We conduct a comprehensive evaluation of our approach on multiple simulated robotic manipulation tasks, comparing its performance against other baseline methods.

II. Preliminaries

A. Episodic Reinforcement Learning

a) Markov Decision Process (MDP).: We consider a MDP problem of a policy search defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \mathcal{P}_0, \gamma)$. We assume the state space $\mathcal{S}$ and action space $\mathcal{A}$ are continuous and the transition probabilities $\mathcal{T} : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$ describe the state transition probability to $\boldsymbol{s}_{t+1}$, given the current state $\boldsymbol{s}_t \in \mathcal{S}$ and action $\boldsymbol{a}_t \in \mathcal{A}$. The initial state distribution is denoted as $\mathcal{P}_0 : \mathcal{S} \to [0, 1]$. The reward $r_t(\boldsymbol{s}_t, \boldsymbol{a}_t)$ returned by the environment is given by a function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and $\gamma \in [0, 1]$ describes the discount factor. The goal of RL in general is to find a policy π that maximizes the expected accumulated reward, namely return, as $R = \mathbb{E}_{\mathcal{T},\mathcal{P}_0,\pi}[\sum_{t=0}^{\infty} \gamma^t r_t]$.

Episodic RL [9] focuses on maximizing the return $R = \sum_{t=0}^{T}[\gamma^t r_t]$ over a task episode of length T, irrespective of the state transitions within the episode. This approach typically employs a parameterized trajectory generator, like MPs [12], to predict a trajectory parameter vector $\boldsymbol{w}$. This vector is then used to generate a complete reference trajectory $\boldsymbol{y}(\boldsymbol{w}) = [y_t]_{t=0:T}$. The resulting trajectory is executed using a trajectory tracking controller to accomplish the task. In this context, $y_t \in \mathbb{R}^D$ denotes the trajectory value at time t for a system with D DoF, differentiating it from the per-step action $\boldsymbol{a}$ used in SRL. It is important to note that, although ERL predicts an entire action trajectory, it still maintains the *Markov Property*, where the state transition probability depends only on the given current state and action [18]. In this respect, the action selection process in ERL is fundamentally similar to techniques like action repeat [19] and temporally correlated action selection [6, 20]. In contrast to SRL, ERL predicts the trajectory parameters as $\pi(\boldsymbol{w}|\boldsymbol{s})$, which shifts the solution search from the per-step action space $\mathcal{A}$ to the parameter space $\mathcal{W}$. Therefore, a trajectory parameterized by a vector $\boldsymbol{w}$ is typically treated as a single data point in $\mathcal{W}$. Consequently, ERL commonly employs black-box optimization methods for problem-solving [14, 17]. The general learning objective of ERL is formally expressed as

$$\begin{aligned} J &= \int \pi_{\boldsymbol{\theta}}(\boldsymbol{w}|\boldsymbol{s})[R(\boldsymbol{s}, \boldsymbol{w}) - V^{\pi}(\boldsymbol{s})]d\boldsymbol{w} \\ &= \mathbb{E}_{\boldsymbol{w}\sim\pi_{\boldsymbol{\theta}}(\boldsymbol{w}|\boldsymbol{s})}[A(\boldsymbol{s}, \boldsymbol{w})], \end{aligned} \tag{1}$$

where $\pi_{\boldsymbol{\theta}}$ represents the policy, parameterized by $\boldsymbol{\theta}$, e. g. using NNs. The initial state $\boldsymbol{s} \in \mathcal{S}$ characterizes the starting configuration of the environment and the task goal, serving as the input to the policy. The $\pi_{\boldsymbol{\theta}}(\boldsymbol{w}|\boldsymbol{s})$ indicates the likelihood of selecting the trajectory parameter $\boldsymbol{w}$. The term $R(\boldsymbol{s}, \boldsymbol{w}) = \sum_{t=0}^{T}[\gamma^t r_t]$ represents the return obtained from executing the trajectory, while $V^{\pi}(\boldsymbol{s}) = \mathbb{E}_{\boldsymbol{w}\sim\pi_{\boldsymbol{\theta}}(\boldsymbol{w}|\boldsymbol{s})}[R(\boldsymbol{s}, \boldsymbol{w})]$ denotes the expected return across all possible trajectories under policy $\pi_{\boldsymbol{\theta}}$. Their subtraction is defined as the advantage function $A(\boldsymbol{s}, \boldsymbol{w})$, which quantifies the benefit of selecting a specific trajectory. By using parameterized trajectory generators like MPs, ERL benefits from consistent exploration, smooth trajectories, and robustness against local optima, as noted by Otto et al. [17]. However, its policy update strategy incurs a trade-off in terms of learning efficiency, as valuable step-based information is overlooked during policy updates. Furthermore, existing method like Otto et al. [17] and Bahl et al. [21] commonly use factorized Gaussian policies, which inherently limits their capacity to capture all relevant movement correlations.

B. Using Movement Primitives for Trajectory Representation

The Movement Primitives (MP), as a parameterized trajectory generator, play an important role in ERL and robot learning. This section highlights key MP methodologies and their mathematical foundations. Schaal [12] introduced the Dynamic Movement Primitives (DMPs) method, incorporating a force signal into a dynamical system to produce

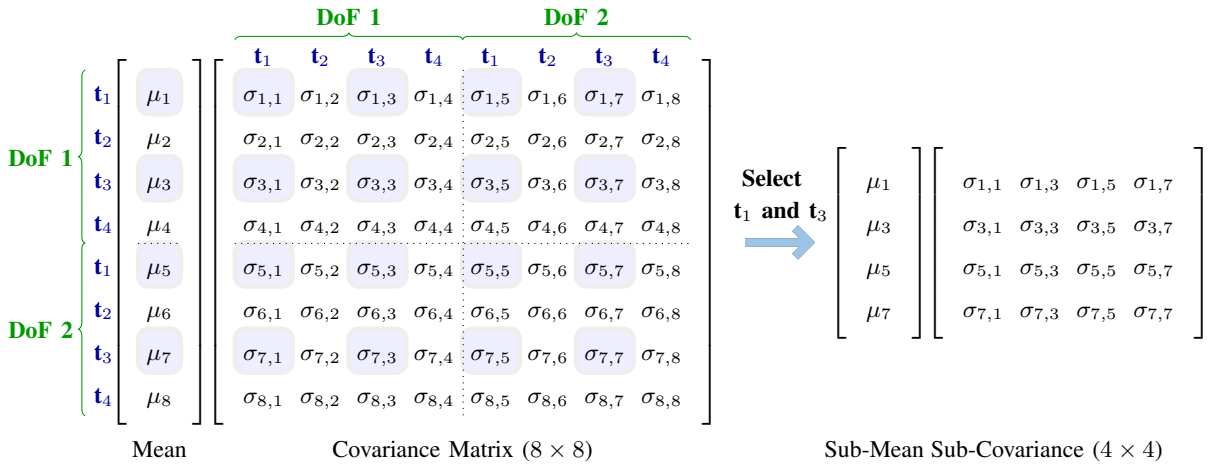


Fig. 2: Reduce the trajectory distribution dimensions using two time steps [22], shown in an element-wise format. Here, the trajectory has two DoF and four time steps, with $D \cdot T = 8$. **Left**: The 8-dim mean vector and the 8×8-dim covariance matrix of the original trajectory distribution, capture correlations across both DoF and time steps. **Right**: Randomly selecting two time points, e. g. t_1 and t_3, yields a reduced distribution while still capturing the movement correlations.

smooth trajectories from given initial robot states. Following this, Paraschos et al. [13] developed Probabilistic Movement Primitives (ProMPs), which leverages a linear basis function representation to map parameter vectors to trajectories and their corresponding distributions. The probability of observing a trajectory $[y_t]_{t=0:T}$ given a specific weight vector distribution $p(\boldsymbol{w}) \sim \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu_w}, \boldsymbol{\Sigma_w})$ is represented as a linear basis function model:

$$[y_t]_{t=0:T} = \boldsymbol{\Phi}_{0:T}^{\intercal}\boldsymbol{w} + \epsilon_y, \tag{2}$$

$$p([y_t]_{t=0:T};\ \boldsymbol{\mu_y}, \boldsymbol{\Sigma_y}) = \mathcal{N}(\boldsymbol{\Phi}_{0:T}^{\intercal}\boldsymbol{\mu_w},\ \boldsymbol{\Phi}_{0:T}^{\intercal}\boldsymbol{\Sigma_w}\boldsymbol{\Phi}_{0:T}\ + \sigma_y^2\boldsymbol{I}). \tag{3}$$

Here, ϵ_y is zero-mean white noise with variance σ_y^2. The matrix $\boldsymbol{\Phi}_{0:T}$ houses the basis functions for each time step t. Additionally, $p([y_t]_{t=0:T};\ \boldsymbol{\mu_y}, \boldsymbol{\Sigma_y})$ defines the trajectory distribution coupling the DoF and time steps, mapped from $p(\boldsymbol{w})$. For a D-DoF system with N parameters per DoF and T time steps, the dimensions of the variables in Eq. (2) and 3 are as follows: $\boldsymbol{w}, \boldsymbol{\mu_w} : D \cdot N$; $\boldsymbol{\Sigma}_w : D \cdot N \times D \cdot N$; $\boldsymbol{\Phi}_{0:T} : D \cdot N \times D \cdot T$; $\boldsymbol{y}, \boldsymbol{\mu_y} : D \cdot T$; $\boldsymbol{\Sigma_y} : D \cdot T \times D \cdot T$.

Recently, Li et al. [22] introduced Probabilistic Dynamic Movement Primitives (ProDMPs), a hybrid approach that blends the pros of both methods. Similar to ProMP, ProDMPs defines a trajectory as $y(t) = \boldsymbol{\Phi}(t)^{\intercal}\boldsymbol{w} + c_1 y_1(t) + c_2 y_2(t)$. The added terms $c_1 y_1(t) + c_2 y_2(t)$ are included to ensure accurate trajectory initialization. This formulation combines the distributional modeling benefits of ProMP with the precision in trajectory initiation offered by DMP.

C. Representation of Trajectory Distribution and Likelihood

Computing the trajectory distribution and reconstruction likelihood is crucial for policy updates in ERL. Previous methods like Otto et al. [17] and Bahl et al. [21] represented the trajectory distribution using the parameter distribution $p(\boldsymbol{w})$ and the likelihood of a sampled trajectory $\boldsymbol{y}^*$ with its parameter vector as $p(\boldsymbol{w}^*|\boldsymbol{\mu_w}, \sigma_{\boldsymbol{w}}^2)$. However, this approach treats an entire trajectory as a singular data point and fails to efficiently utilize step-based information. In contrast, research in imitation learning, including works by Paraschos et al. [13] and Gomez-Gonzalez et al. [23], maps parameter distributions to trajectory space and allows the exploitation of trajectory-specific information. Yet, the likelihood computation in this space is computationally intensive, primarily due to the need to invert a high-dimensional covariance matrix, a process with an $O((D \cdot T)^3)$ time complexity. Recent studies, like those by [24–26], advocates for directly modeling the trajectory distribution using neural networks. These methods typically employ a factorized Gaussian distribution $\mathcal{N}(y|\mu_y, \sigma_y^2)$, instead of a full Gaussian distribution $\mathcal{N}(\boldsymbol{y}|\boldsymbol{\mu_y}, \boldsymbol{\Sigma_y})$ that accounts for both the DoF and time steps. This choice mitigates the computational burden of likelihood calculations, but comes at the cost of neglecting key temporal correlations and interactions between different DoF. To address these challenges, Li et al. [22] introduced a novel approach for estimating the trajectory likelihood with a set of paired time points $(t_k, t'_k), k = 1, ..., K$, as

$$\log p([\boldsymbol{y}_t]_{t=0:T}) \approx \frac{1}{K}\sum_{k=1}^{K} \log \mathcal{N}(\boldsymbol{y}_{(t_k,t'_k)}|\boldsymbol{\mu}_{(t_k,t'_k)}, \boldsymbol{\Sigma}_{(t_k,t'_k)}), \tag{4}$$

As shown in Fig. 2, this method scales down the dimensions of a trajectory distribution from $D \cdot T$ to a more manageable $D \cdot 2$. Through the use of batched, randomly selected time pairs during training, the method is proved to efficiently capture correlations while reducing computational cost.

D. Using Trust Regions for stable policy update

In ERL, the parameter space $\mathcal{W}$ typically exhibits higher dimensionality compared to the action space $\mathcal{A}$. This complexity presents unique challenges in maintaining stable

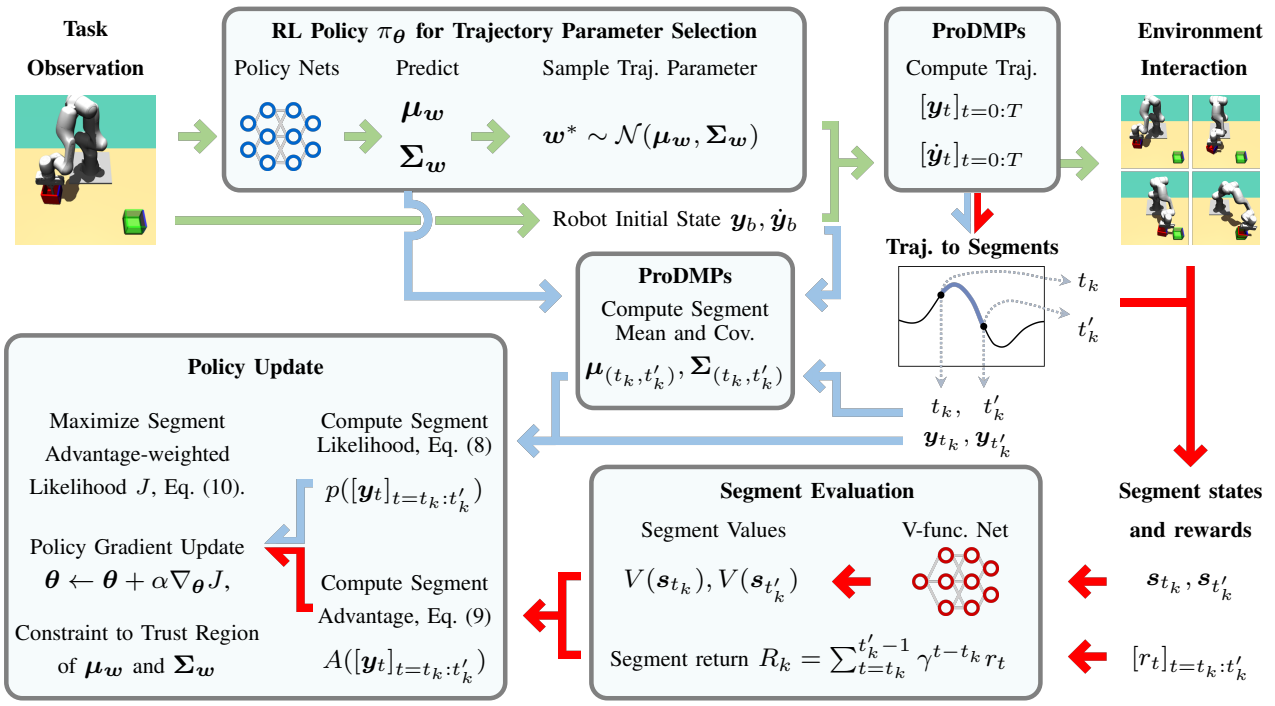


Fig. 3: The TCE framework. The entire learning framework can be divided into three main parts. The first part, shown in green arrows, involves trajectory sampling, generation, and execution, detailing how the robot is controlled to complete a given task. The second part, indicated in blue arrows, focuses on estimating the likelihood of selecting a particular segment of the sampled trajectory. The third part, marked by red arrows, deals with segment evaluation and advantage computation, assessing how much each segment contributes to the successful task completion.

policy updates. Trust regions methods [1, 27] has long been recognized as an effective technique for ensuring the stability and convergence of policy gradient methods. While popular methods such as PPO approximate trust regions using surrogate cost functions, they lack the capacity for exact enforcement. To tackle this issue, Otto et al. [28] introduced trust region projection layer (TRPL), a mathematically rigorous and scalable technique that precisely enforces trust regions in deep RL algorithms. By incorporating differentiable convex optimization layers [29], this method not only allows for trust region enforcement for each input state, but also demonstrates significant effectiveness and stability in high-dim parameter space, as validated in method like BBRL [17]. The TRPL takes standard outputs of a Gaussian policy—namely, the mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ —and applies a state-specific projection operation to maintain trust regions. The adjusted Gaussian policy, parameterized by $\tilde{\boldsymbol{\mu}}$ and $\tilde{\boldsymbol{\Sigma}}$, forms the basis for subsequent computations. Let d_{mean} and d_{cov} be the dissimilarity measures, e. g. KL-divergence, for mean and covariance, bounded by ϵ_μ and ϵ_Σ respectively. The optimization for each state $\boldsymbol{s}$ is formulated as:

$$\begin{aligned} &\arg\min_{\tilde{\boldsymbol{\mu}}_s} d_{\text{mean}}\left(\tilde{\boldsymbol{\mu}}_s, \boldsymbol{\mu}(\boldsymbol{s})\right), \quad \text{s.t.} \quad d_{\text{mean}}\left(\tilde{\boldsymbol{\mu}}_s, \boldsymbol{\mu}_{\text{old}}(\boldsymbol{s})\right) \leq \epsilon_\mu, \\ &\arg\min_{\tilde{\boldsymbol{\Sigma}}_s} d_{\text{cov}}\left(\tilde{\boldsymbol{\Sigma}}_s, \boldsymbol{\Sigma}(\boldsymbol{s})\right), \quad \text{s.t.} \quad d_{\text{cov}}\left(\tilde{\boldsymbol{\Sigma}}_s, \boldsymbol{\Sigma}_{\text{old}}(\boldsymbol{s})\right) \leq \epsilon_\Sigma. \end{aligned} \tag{5}$$

III. Use Step-based Information for ERL Policy Updates

We introduce an innovative framework of ERL that builds on traditional ERL foundations, aiming to facilitate an efficient policy update mechanism while preserving the intrinsic benefits of ERL. The key innovation lies in redefining the role of trajectories in the policy update process. In contrast to previous methods which consider an entire trajectory as a single data point, our approach breaks down the trajectory into individual segments. Each segment is evaluated and weighted based on its distinct contribution to the task success. This method allows for a more effective use of step-based information in ERL. The comprehensive structure of this framework is depicted in Figure 3.

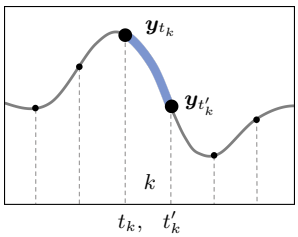


Fig. 4: Divide a trajectory into K segments

Trajectory Prediction and Generation. As highlighted by green arrows in Fig. 3, we adopt a structure similar to previous ERL works, such as the one described by Otto et al. [17]. However, this part distinguish itself by using the most recent ProDMPs for trajectory generation and distri-

bution modeling, due to the improved support for trajectory initialization. Additionally, we enhance the previous framework by using a full covariance matrix policy $\pi(\boldsymbol{w}|\boldsymbol{s}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{\mu_w}, \boldsymbol{\Sigma_w})$ as opposed to a factorized Gaussian policy, to capture a broader range of movement correlations.

Trajectory Likelihood Representation. In RL, the likelihood of previously sampled actions, along with their associated returns, is often used to adjust the chance of these actions being selected in future policies. In previous ERL methods, this process typically involves the probability of choosing an entire trajectory. However, our framework adopts a different strategy, as shown in blue arrows in Fig. 3. Using the techniques in Sections II-B and II-C, our approach begins by selecting K paired time steps. We then transform the parameter likelihood into a trajectory likelihood, which is calculated using these K pairwise likelihoods. This approach, depicted in Figure 4, effectively divides the whole trajectory into K distinct segments, with each segment defined by a pair of time steps. In essence, this method allows us to break down the overall trajectory likelihood into individual segment likelihoods, offering a more detailed view of the trajectory's contribution to task success.

Trajectory to Segments:

$$p([\boldsymbol{y}_t]_{t=0:T}|\boldsymbol{s}) \triangleq \{p([\boldsymbol{y}_t]_{t=t_k:t'_k}|\boldsymbol{s})\}_{k=1\dots K}, \tag{6}$$

(7)

Local Representation:

$$p([\boldsymbol{y}_t]_{t=t_k:t'_k}|\boldsymbol{s}) \triangleq p([\boldsymbol{y}_{t_k}, \boldsymbol{y}_{t'_k}]|\boldsymbol{\mu}_{(t_k,t'_k)}(\boldsymbol{s}), \boldsymbol{\Sigma}_{(t_k,t'_k)}(\boldsymbol{s})). \tag{8}$$

Definition of Segment Advantages. Similar to standard SRL methods, we leverage the advantage function to evaluate the benefits of executing individual segments within a sampled trajectory. When being at state $\boldsymbol{s}_{t_k}$ and following the trajectory segment $[\boldsymbol{y}_t]_{t=t_k:t'_k}$, the segment-wise advantage function $A(\boldsymbol{s}_{t_k}, [\boldsymbol{y}_t]_{t=t_k:t'_k})$ quantifies the difference between the actual return obtained by executing this sampled trajectory segment and the expected return from a randomly chosen segment, as

$$A(\boldsymbol{s}_{t_k}, [\boldsymbol{y}_t]_{t=t_k:t'_k}) = \sum_{t=t_k}^{t=t'_k-1} \gamma^{t-t_k} r_t + \gamma^{t'_k-t_k} V^{\pi_{\text{old}}}(\boldsymbol{s}_{t'_k}) - V^{\pi_{\text{old}}}(\boldsymbol{s}_{t_k}), \tag{9}$$

where $V^{\pi_{\text{old}}}(\vec{s}_{t_k})$ denotes the value function of the current policy. In our method, the estimation of $V^{\pi_{\text{old}}}(\vec{s}_{t_k})$ is consistent with the approaches commonly used in SRL and is independent of the design choice of segment selections. We use NNs to estimate $V^{\pi}(\boldsymbol{s}) \approx V^{\pi}_{\phi}(\boldsymbol{s})$ which is fitted on targets of true return or obtained by general advantage estimation (GAE) [5].

Learning Objective. By replacing the trajectory likelihood and advantage with their segment-based counterparts in the original ERL learning objective as stated in Eq. (1), we propose the learning objective of our method as follows

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\pi_{\text{old}}}\left[\frac{1}{K}\sum_{k=1}^{K} \frac{p_{\pi_{\text{new}}}([\boldsymbol{y}_t]_{t=t_k:t'_k}|\boldsymbol{s})}{p_{\pi_{\text{old}}}([\boldsymbol{y}_t]_{t=t_k:t'_k}|\boldsymbol{s})} A^{\pi_{\text{old}}}(\boldsymbol{s}_{t_k}, [\boldsymbol{y}_t]_{t=t_k:t'_k})\right]. \tag{10}$$

Here, $\boldsymbol{s}$ denotes the initial state of the episode, used for selecting the parameter vector $\vec{w}$, and $\boldsymbol{s}_{t_k}$ is the state of the system at time t_k. The learning objective takes the *Importance Sampling* to update policies using data from previous policies [1, 27, 28]. Our method retains the advantages of exploration in parameter space and generating smooth trajectories. This enables us to enhance the likelihood of segments with high advantage and reduce the likelihood of less rewarding ones during policy updates. To ensure a stable update for the full covariance Gaussian policy $\pi_{\boldsymbol{\theta}}(\boldsymbol{w}|\boldsymbol{s}) = \mathcal{N}(\boldsymbol{\mu_w}, \boldsymbol{\Sigma_w})$, we deploy a differentiable Trust Region Projection step [28] after each policy update iteration as previously discussed in Section II-D.

IV. Related Works

Improve Exploration and Smoothness in Step-based RL. SRL methods, such as PPO and SAC, interact with the environment by performing a sampled action at each time-step. This strategy often results in a control signal with high-frequency noise, making it unsuitable for direct use in robotic systems [6]. A prevalent solution is to reduce the sampling frequency, a technique commonly known as *frame skip* [19]. Here, the agent only samples actions every k-th time step and replicates this action for the skipped steps. Similar approaches decide whether to repeat the last action or to sample a new action in every time step [30, 31]. This concept is also echoed in works such as general State Dependent Exploration (gSDE) [6, 32, 33], where the applied noise is sampled in a state-dependent fashion; leading to smooth changes of the disturbance between consecutive steps. However, while these methods improved the smoothness in small segments, they struggled to model long-horizon correlations. Another area of concern is the utilization of white noise during sampling, which fails to consider the temporal correlations between time steps and results in a random walk with suboptimal exploration. To mitigate this, previous research, such as [34] and [20], have integrated colored noise into the RL policy, aiming to foster exploration that is correlated across time steps. While these methods have shown advantages over white noise approaches, they neither improve the trajectory's smoothness, nor adequately capture the cross-DoF correlations.

Episodic RL. The early ERL approaches used black-box optimization to evolve parameterized policies, e.g., small NN [9, 10, 35]. However, these early works were limited to tasks with low-dimensional action space, for instance, the Cart Pole. Although recent works [14, 36] have shown that, with more computing, these methods can achieve comparable asymptotic performance with step-based algorithms in some locomotion tasks, none of these methods can deal with

tasks with context variations (e.g., changing goals). Another research line in ERL works with more compact policy representation. [11, 37] first combined ERL with MPs, reducing the dimension of searching space from NN parameter space to MPs parameter space with the extra benefits of smooth trajectories generation. [38] proposed a model-based method to improve the sample efficiency. Notably, although those methods can already handle some complex manipulation tasks such as robot baseball [11], none of them can deal with contextual tasks. To deal with that problem, [39] further extends that utilizes linear policy conditioned on the context. Another recent work from this research line [16] proposed using a Mixture of Experts (MoE) to learn versatile skills under the ERL framework.

BBRL. As the first method that utilizes non-linear adaptation to contextual ERL, Deep Black Box Reinforcement Learning (BBRL) [17] is the most related work to our method. BBRL applies trust-region-constrained policy optimization to learn a weight adaptation policy for MPs. Despite demonstrating great success in learning tasks with sparse and non-Markovian rewards, it requires significantly more samples to converge compared to SoTA SRL methods. This could be attributed to its black-box nature, where the trajectory from the entire episode is treated as a single data point, and the trajectory return is calculated by summing up all step rewards within the episode.

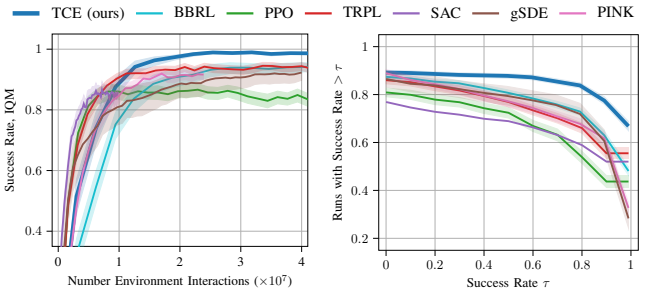
V. Experiments

We evaluate the effectiveness of our model through experiments on a variety of simulated robot manipulation tasks. The performance of TCE is compared against well-known deep RL algorithms as well as methods specifically designed for correlated actions and consistent trajectories. The evaluation is designed to answer the following questions: (a) Can our model effectively train the policy across diverse tasks, incorporating various robot types, control paradigms (task and joint space), and reward configurations? (b) Does the incorporation of movement correlations lead to higher task success rates? (c) Are there limitations or trade-offs when adopting our proposed learning strategy?

For the comparative evaluation, we select the following methods: PPO, SAC, TRPL, gSDE, PINK [20] and BBRL. We use step-based methods (PPO, SAC, TRPL, gSDE, and PINK) to predict the lower-level actions for each task. On the other hand, for episodic methods like BBRL and TCE, we predict position and velocity trajectories and then employ a PD controller to compute the lower-level control commands. Across all experiments, we measure task success in terms of the number of environment interactions required. Each algorithm is evaluated using 20 distinct random seeds. Results are quantified using the Interquartile Mean (IQM) and are accompanied by a 95% stratified bootstrap confidence interval [8].

A. Large Scale Robot Manipulation Benchmark

We begin our evaluation using the Metaworld benchmark [40], a comprehensive testbed that includes 50 manipulation tasks of varying complexity. Control is executed in a 3-DOF task space along with the finger closeness, and a dense reward signal is employed. In contrast to the original evaluation protocol, we introduce a more stringent framework. Specifically, each episode is initialized with a randomly generated context, rather than a fixed one. Additionally, we tighten the success criteria to only consider a task successfully completed if the objective is maintained until the final time step. This adjustment mitigates scenarios where transient successes are followed by erratic agent behavior. While we train separate policies for each task, the hyperparameters remain constant across all 50 tasks. For each method, we report the overall success rate as measured by the IQM across the 50 sub-tasks in Fig. 5a. The performance profiles are presented in Fig. 5b.



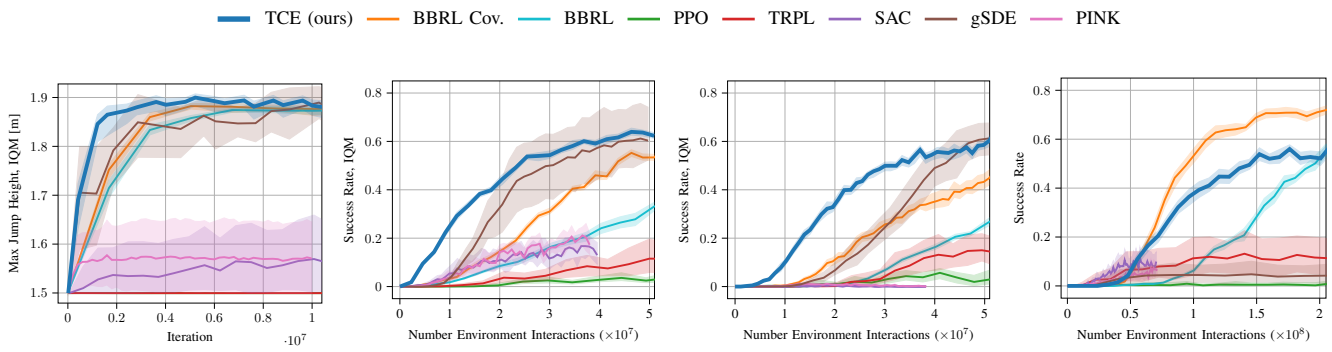
(a) Success Rate (b) Performance Profile

Fig. 5: Metaworld Evaluation. (a) Overall Success Rate across all 50 tasks, reported using Interquartile Mean (IQM) [8]. (b) Performance profile, illustrating the fraction of runs that exceed the threshold specified on the x-axis.

In both metrics, our method significantly outperforms the baselines in achieving task success. BBRL exhibits the second-best performance in terms of overall consistency across tasks but lags in training speed compared to step-based methods. We attribute this difference to the use of per-step dense rewards, which enables faster policy updates in step-based approaches. TCE leverages the advantages of both paradigms, surpassing other algorithms after approximately 10^7 environment interactions. Notably, the off-policy methods SAC and PINK were trained with fewer samples than used for on-policy methods due to their limitations in parallel environment utilization. PINK achieved superior final performance but at the expense of sample efficiency compared to SAC.

B. Joint Space Control with Multi Task Objectives

Next, we investigate the advantages of modeling complete movement correlations and the utility of intermediate feedback for policy optimization. To this end, we enhance the BBRL algorithm by expanding its factorized Gaussian policy to accommodate full covariance (BBRL Cov.), thereby capturing movement correlations. Both the original and augmented versions of BBRL are included in the subsequent task evaluations. We evaluate the methods on a customized Hopper Jump task, sourced from OpenAI Gym [41]. This 3-



(a) Hopper Jump, Max Height (b) Box Pushing, Dense (c) Box Pushing, Sparse (d) Table Tennis

Fig. 6: Task Evaluation of (a) Hopper Jump Max Height. (b) Box Pushing success rate in dense reward, and (c) Box Pushing success rate in sparse reward setting. (d) Table tennis with high reward sparsity.

DoF task primarily focuses on maximizing jump height while also accounting for precise landing at a designated location. Control is executed in joint space. We report the max jump height as the main metric of success in Fig. 6a. Our method exhibits quick learning and excels in maximizing jump height. Both BBRL versions exhibit similar performance, while BBRL Cov. demonstrates marginal improvements over the original. However, they both fall behind TCE in training speed, highlighting the efficiency gains we achieve through intermediate state-based policy updates. Step-based methods like PPO and TRPL tend to converge to sub-optimal policies. The only exception is gSDE. As an augmented step-based method, it enables smoother and more consistent exploration but exhibits significant sensitivity to model initialization (random seeds), evident from the wide confidence intervals.

C. Contact-rich Manipulation with Dense and Sparse Reward Settings

We further turn to a 7-DoF robot box-pushing task adapted from [17]. The task requires the robot's end-effector, equipped with a rod, to maneuver a box to a specified target position and orientation. The difficulty lies in the need for continuous, correlated movements to both position and orient the box accurately. To amplify the complexity, the initial pose of the box is randomized. We test two reward settings: dense and sparse. The dense setting offers intermediate rewards inversely proportional to the current distance between the box and its target pose, while the sparse setting only allocates rewards at the episode's end based on the final task state. Performance metrics for both settings are shown in Fig. 6b and 6c. In either case, TCE and gSDE exhibit superior performance but with TCE demonstrating greater consistency across different random seeds. The augmented BBRL version outperforms its original counterpart, emphasizing the need for fully correlated movements in tasks that demand consistent object manipulation. The other step-based methods struggle to learn the task effectively, even when dense rewards are provided. This further highlights the advantages of modeling the movement trajectory as a unified action, as opposed to a step-by-step approach.

D. Hitting Task with High Sparsity Reward Setting

In our last experiment, we assess the limitations of our method using a 7-DoF robot table tennis task, originally from [16]. The robot aims to return a randomly incoming ball to a desired target on the opponent's court. To enhance the task's realism, we randomize the robot's initial state instead of using a fixed starting pose. This task is distinct due to its one-shot nature: the robot has only one chance to hit the ball and loses control over the ball's trajectory thereafter. The need for diverse hitting strategies like forehand and backhand adds complexity and increases the number of samples required for training. Performance metrics are presented in Fig. 6d. The BBRL Cov. emerges as the leader, achieving a 20% higher success rate than other methods. It is followed by TCE and the original BBRL, with TCE displaying intermediate learning speeds between the two BBRL versions. Step-based methods, led by TRPL at a mere 15% task success, struggle notably in this setting. We attribute the underperformance of TCE and step-based methods to the task's reward sparsity, which complicates the value function learning of SRL and TCE. Despite these challenges, TCE maintains its edge over other baselines, further attesting to its robustness, even under stringent conditions.

VI. Conclusion

We introduced TCE that synergizes the exploration advantages of ERL with the sample efficiency of SRL. Empirical evaluation showcases that TCE matches the sample efficiency of SRL and consistently delivers competitive asymptotic performance across various tasks. Furthermore, we demonstrated both the sample efficiency and policy performance of episodic policies can be further improved by incorporating proper correlation modeling. Several opening questions remain for future work. TCE yields moderate results for tasks characterized by particularly sparse reward settings, as observed in scenarios like table tennis. Additionally, the current open-loop control setup lacks the adaptability needed for complex control problems in dynamic environments where immediate feedback and swift adaptation are crucial. These issues will be at the forefront of our future work.

VII. Acknowledgement and Statement

This paper is a workshop version of the original paper [42], where more technical details can be found. The authors acknowledge support by the state of Baden-Württemberg, Germany, through bwHPC, and the HoreKa supercomputer.

References

[1] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347* (2017).

[2] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor". In: *International conference on machine learning*. PMLR. 2018, pp. 1861–1870.

[3] R. J. Williams. "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Mach. learning* 8 (1992), pp. 229–256.

[4] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. "Policy gradient methods for reinforcement learning with function approximation". In: *Adv. neural information processing systems* 12 (1999).

[5] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. "High-dimensional continuous control using generalized advantage estimation". In: *arXiv preprint arXiv:1506.02438* (2015).

[6] A. Raffin, J. Kober, and F. Stulp. "Smooth exploration for robotic reinforcement learning". In: *Conference on Robot Learning*. PMLR. 2022, pp. 1634–1644.

[7] P. Schumacher, D. F. Haeufle, D. Büchler, S. Schmitt, and G. Martius. "DEP-RL: Embodied Exploration for Reinforcement Learning in Overactuated and Musculoskeletal Systems". In: *Proceedings of the Eleventh International Conference on Learning Representations (ICLR)*. May 2023.

[8] R. Agarwal, M. Schwarzer, P. S. Castro, A. Courville, and M. G. Bellemare. "Deep Reinforcement Learning at the Edge of the Statistical Precipice". In: *Adv. Neural Inf. Process. Syst.* (2021).

[9] D. Whitley, S. Dominic, R. Das, and C. W. Anderson. "Genetic reinforcement learning for neurocontrol problems". In: *Mach. Learn.* 13 (1993), pp. 259–284.

[10] C. Igel. "Neuroevolution for reinforcement learning using evolution strategies". In: *The 2003 Congress on Evolutionary Computation, 2003. CEC'03.* Vol. 4. IEEE. 2003, pp. 2588–2595.

[11] J. Peters and S. Schaal. "Reinforcement learning of motor skills with policy gradients". In: *Neural networks* 21.4 (2008), pp. 682–697.

[12] S. Schaal. "Dynamic movement primitives-a framework for motor control in humans and humanoid robotics". In: *Adaptive motion of animals and machines*. Springer, 2006, pp. 261–280.

[13] A. Paraschos, C. Daniel, J. Peters, and G. Neumann. "Probabilistic movement primitives". In: *Adv. neural information processing systems* 26 (2013).

[14] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever. "Evolution strategies as a scalable alternative to reinforcement learning". In: *arXiv preprint arXiv:1703.03864* (2017).

[15] V. Tangkaratt, H. Van Hoof, S. Parisi, G. Neumann, J. Peters, and M. Sugiyama. "Policy search with high-dimensional context variables". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 31. 1. 2017.

[16] O. Celik, D. Zhou, G. Li, P. Becker, and G. Neumann. "Specializing versatile skill libraries using local mixture of experts". In: *Conference on Robot Learning*. PMLR. 2022, pp. 1423–1433.

[17] F. Otto, O. Celik, H. Zhou, H. Ziesche, V. A. Ngo, and G. Neumann. "Deep black-box reinforcement learning with movement primitives". In: *Conference on Robot Learning*. PMLR. 2022, pp. 1244–1265.

[18] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[19] A. Braylan, M. Hollenbeck, E. Meyerson, and R. Miikkulainen. "Frame skip is a powerful parameter for learning to play atari". In: *Workshops at the twenty-ninth AAAI conference on artificial intelligence*. 2015.

[20] O. Eberhard, J. Hollenstein, C. Pinneri, and G. Martius. "Pink noise is all you need: Colored noise exploration in deep reinforcement learning". In: *The Eleventh International Conference on Learning Representations*. 2022.

[21] S. Bahl, M. Mukadam, A. Gupta, and D. Pathak. "Neural dynamic policies for end-to-end sensorimotor learning". In: *Adv. Neural Inf. Process. Syst.* 33 (2020), pp. 5058–5069.

[22] G. Li, Z. Jin, M. Volpp, F. Otto, R. Lioutikov, and G. Neumann. "ProDMP: A Unified Perspective on Dynamic and Probabilistic Movement Primitives". In: *IEEE Robotics Autom. Lett.* 8.4 (2023), pp. 2325–2332.

[23] S. Gomez-Gonzalez, G. Neumann, B. Schölkopf, and J. Peters. "Using probabilistic movement primitives for striking movements". In: *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE. 2016, pp. 502–508.

[24] M. Y. Seker, M. Imre, J. H. Piater, and E. Ugur. "Conditional Neural Movement Primitives." In: *Robotics: Science and Systems*. Vol. 10. 2019.

[25] M. Akbulut, E. Oztop, M. Y. Seker, X. Hh, A. Tekden, and E. Ugur. "Acnmp: Skill transfer and task extrapolation through learning from demonstration and reinforcement learning via representation sharing". In: *Conference on Robot Learning*. PMLR. 2021, pp. 1896–1907.

[26] M. Przystupa, F. Haghverd, M. Jagersand, and S. Tosatto. "Deep Probabilistic Movement Primitives with a Bayesian Aggregator". In: *arXiv preprint arXiv:2307.05141* (2023).

[27] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. "Trust region policy optimization". In: *International conference on machine learning*. PMLR. 2015, pp. 1889–1897.

[28] F. Otto, P. Becker, N. A. Vien, H. C. Ziesche, and G. Neumann. "Differentiable trust region layers for deep reinforcement learning". In: *Int. Conf. on Learn. Represent.* (2021).

[29] A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, and J. Z. Kolter. "Differentiable convex optimization layers". In: *Adv. neural information processing systems* 32 (2019).

[30] A. Biedenkapp, R. Rajan, F. Hutter, and M. Lindauer. "TempoRL: Learning when to act". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 914–924.

[31] H. Yu, W. Xu, and H. Zhang. "Taac: Temporally abstract actor-critic for continuous control". In: *Adv. Neural Inf. Process. Syst.* 34 (2021), pp. 29021–29033.

[32] T. Rückstieß, M. Felder, and J. Schmidhuber. "State-Dependent Exploration for Policy Gradient Methods". In: *Machine Learning and Knowledge Discovery in Databases*. Ed. by W. Daelemans, B. Goethals, and K. Morik. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 234–249.

[33] A. S. Chiappa, A. M. Vargas, A. Z. Huang, and A. Mathis. "Latent exploration for reinforcement learning". In: *Adv. Neural Inf. Process. Syst. (NeurIPS)* (2023).

[34] T. P. Lillicrap et al. "Continuous control with deep reinforcement learning". In: *arXiv preprint arXiv:1509.02971* (2015).

[35] F. Gomez, J. Schmidhuber, R. Miikkulainen, and M. Mitchell. "Accelerated Neural Evolution through Cooperatively Coevolved Synapses." In: *J. Mach. Learn. Research* 9.5 (2008).

[36] H. Mania, A. Guy, and B. Recht. "Simple random search of static linear policies is competitive for reinforcement learning". In: *Adv. Neural Inf. Process. Syst.* 31 (2018).

[37] J. Kober and J. Peters. "Policy search for motor primitives in robotics". In: *Adv. neural information processing systems* 21 (2008).

[38] A. Abdolmaleki, R. Lioutikov, J. R. Peters, N. Lau, L. Pualo Reis, and G. Neumann. "Model-based relative entropy stochastic search". In: *Adv. Neural Inf. Process. Syst.* 28 (2015).

[39] A. Abdolmaleki, B. Price, N. Lau, L. P. Reis, and G. Neumann. "Contextual covariance matrix adaptation evolutionary strategies". In: International Joint Conferences on Artificial Intelligence Organization (IJCAI). 2017.

[40] T. Yu et al. "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning". In: *Conference on robot learning*. PMLR. 2020, pp. 1094–1100.

[41] G. Brockman et al. "Openai gym". In: *arXiv preprint arXiv:1606.01540* (2016).

[42] G. Li et al. "Open the Black Box: Step-based Policy Updates for Temporally-Correlated Episodic Reinforcement Learning". In: *The Twelfth International Conference on Learning Representations*. 2024.