Discovering Diverse Nearly Optimal Policies with Successor Features

Abstract

Finding different solutions to the same problem is a key aspect of intelligence associated with creativity and adaptation to novel situations. In reinforcement learning, a set of diverse policies can be useful for exploration, transfer, hierarchy, and robustness. We propose Diverse Successive Policies, a method for discovering policies that are diverse in the space of Successor Features, while assuring that they are near optimal. We formalize the problem as a Constrained Markov Decision Process (CMDP) where the goal is to find policies that maximize diversity, characterized by an intrinsic diversity reward, while remaining near-optimal with respect to the extrinsic reward of the MDP. We also analyze how recently proposed robustness and discrimination rewards perform and find that they are sensitive to the initialization of the procedure and may converge to sub-optimal solutions. To alleviate this, we propose new explicit diversity rewards that aim to minimize the correlation between the Successor Features of the policies in the set. We compare the different diversity mechanisms in the DeepMind Control Suite and find that the type of explicit diversity we are proposing is important to discover distinct behavior, like for example different locomotion patterns.

1. Introduction

Creative problem solving is the mental process of searching for an original and previously unknown solution to a problem (Osborn, 1953). The relationship between creativity and intelligence is widely recognized across many fields; for example, in the field of Mathematics, finding different proofs to the same theorem is considered elegant and often leads to new insights.

Closer to Artificial Intelligence (AI), consider the field of game playing and specifically the game of Chess in which a move is considered creative when it goes beyond known patterns (da Fonseca-Wollheim, 2020). In some cases, such moves can only be detected by human players while remaining invisible to currently state-of-the-art Chess engines. A famous example thereof is the winning move in game eight of the Classical World Chess Championship 2004 between Leko and Kramnik (Behovits, 2004). Humans and indeed many animals employ similarly creative behavior on a daily basis; faced with a challenging problem we often consider qualitatively different alternative solutions.

Yet, the majority of AI research is focused on finding a single best solution to a given problem. For example, in the field of Reinforcement Learning (RL), most algorithms are designed to find a single reward-maximizing policy. However, for many problems of interest there may be many qualitatively different optimal or near-optimal policies; finding such diverse set of policies may help an RL agent become more robust to changes in the task and/or environment and to generalize better to future tasks.

In the field of Quality-Diversity (QD), evolutionary algorithms are used to find useful diverse policies (e.g., (Pugh et al., 2016; Mouret and Clune, 2015; Hong et al., 2018; Masood and Doshi-Velez, 2019; Parker-Holder et al., 2020; Gangwani et al., 2020; Peng et al., 2020; Zhang et al., 2019)). In a related line of work, intrinsic rewards are used to find diverse skills for fast adaptation (Gregor et al., 2017; Eysenbach et al., 2019) to be robust to model miss-specification (Kumar et al., 2020; Zahavy et al., 2020b) and for exploration (Agarwal et al., 2020). It was also suggested that policies that maximize diversity are more correlated with human behaviour than those that maximize only the extrinsic reward (Matusch et al., 2020).

This work makes the following contributions. **First,** we propose an incremental method for discovering a diverse set of near-optimal policies. Each policy in the set is trained to solve a Constrained Markov Decision Process (CMDP). The main objective in the CMDP is to maximize the diversity of the growing set, measured in the space of Successor Features (SFs; Barreto et al., 2017), and the constraint is that the policies are near-optimal. **Second,** we analyze how previously proposed robustness and discrimination mechanisms for the "no-reward" setting perform in terms of diversity in our setup. We find that they are sensitive to the initialization of the procedure and may converge to sub-optimal solutions. To alleviate this, we propose two explicit diversity rewards that aim to minimize the correlation between the SFs of the policies in the set. **Third,** we demonstrate our method in the DeepMind Control Suite (Tassa et al., 2018). Given an extrinsic reward (e.g. for standing or walking) our method discovers qualitatively diverse locomotion behaviours for approximately maximizing this reward.

2. Preliminaries and Notation

An MDP (Puterman, 1984) is a tuple $M \triangleq (\mathcal{S}, \mathcal{A}, P, r, \gamma, \rho)$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $P = \{P^a \mid a \in \mathcal{A}\}$ is the set of transition kernels, $\gamma \in [0, 1)$ is the discount factor and ρ is the initial state distribution. The function $r : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ defines the rewards. A policy in M, denoted by π, is a mapping $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$, where $\mathcal{P}(\mathcal{A})$ is the probability distributions over $\mathcal{A}$.

Usually in RL, the agent's objective is to maximize the expected cumulative extrinsic reward. In this work, we will also be interested in discovering and maximizing intrinsic reward functions (Singh et al., 2010). These rewards can be a function of the policy (e.g., its entropy) or a function of observed features. Let $\boldsymbol{\phi}(s,a) \in [0,1]^d$ be an observable vector of bounded features. Then there is a set of rewards induced by all possible linear combinations of the features $\boldsymbol{\phi}$. Specifically, for any $\mathbf{w} \in \mathbb{R}^d$, we can define a reward function $r_{\mathbf{w}}(s,a) = \mathbf{w} \cdot \boldsymbol{\phi}(s,a)$. Given $\mathbf{w}$, the intrinsic reward $r_{\mathbf{w}}$ is well defined and we will use $\mathbf{w}$ and $r_{\mathbf{w}}$ interchangeably to refer to it. Any policy induces a state transition matrix P^π, where $P^\pi(x,y) = P^{\pi(x)}(x,y)$ is the probability of transitioning from state x to state y when the action is selected according to $\pi(x)$. Thus, any policy yields a Markov chain $(\mathcal{S}, P^\pi)$. By looking at the Markov chain induced by a policy we can study its long-term behavior, such as its stationary distribution. This in turn allows us to define a notion of diversity based on the limiting behavior of policies, in contrast with most previous work on diversity that focus on short-term behavior (Gregor et al., 2017; Eysenbach et al., 2019).

Concretely, in defining diversity we use measures defined in the **average-case** setting. The **stationary distribution** d_π of a Markov chain with transition matrix P^π is defined to be $d_\pi(s) = \lim_{t\to\infty} \Pr(s_t = s | s_0 \sim \rho, \pi)$, which we assume exists and is independent of s_0 for all policies. In *ergodic MDPs* this limit is unique and is known to be the probability distribution satisfying $d_\pi^\top = d_\pi^\top P^\pi$ (Puterman, 1984). The asymptotic average reward value, hereafter simply value, of a policy π under reward function r, denoted v_r^π, can be defined as an expectation over d_π as: $v_r^\pi = \mathbb{E}_{s\sim d_\pi} r(s,\pi(s)) = d_\pi \cdot r_\pi$, where r_π is a vector with $\mathbb{E}_{a\sim\pi(s)} r(s,a)$ in its coordinates. A natural time scale in this long-term average-case context is the **mixing time** of the policy – the time until the Markov chain is "close" to its stationary state distribution. Formally, the ϵ-mixing time T_{mix} of an ergodic Markov chain with a stationary distribution d_π is the smallest time t such that $\forall x_0,\ \text{TV}[\Pr_t(\cdot|x_0), d_\pi] \le \epsilon$, where $\Pr_t(\cdot|x_0)$ is the distribution over states after t steps, starting from x_0 and $\text{TV}[\cdot,\cdot]$ is the total variation distance. In other words, if we follow a policy in an MDP for T_{mix} steps, we will observe states that are approximately distributed according to d_π.

Similarly, we can define the expected features, also known as **successor features**, under d_π as $\psi^\pi = \mathbb{E}_{x\sim d_\pi} \phi(x,\pi(x))$. Note that the SFs are conditioned on ρ and π and that they are vectors in $\mathbb{R}^d$; similar definitions were suggested in (Mehta et al., 2008; Zahavy et al., 2020a). For linear rewards there is a simple way to express the average reward value of the policy (Section 2) using the SFs: $v_w^\pi = \psi^\pi \cdot w$. To keep the notation simple, we will refer to the SFs of policy π^i as ψ^i; and, since we are dealing with different intrinsic rewards, we will use the notation v_d^i to refer to the value of policy π^i for reward r_d.

3. Discovering diverse near-optimal policies

We are interested in discovering a set of n near-optimal policies $\Pi^n = \{\pi^i\}_{i=1}^n$ that are maximally diverse according to some diversity metric. Let Ψ^n be the set of SFs corresponding to the policies in Π^n, then we are interested in solving the following constrained optimization problem:

$$\max_{\Pi^n} \ \text{Diversity}(\Psi^n) \ \text{ s.t } \ \ v_e^\pi \ge \alpha v_e^*, \ \forall \pi \in \Pi^n, \tag{1}$$

where Diversity : $\{\mathbb{R}^d\}^n \to \mathbb{R}$ measures the diversity of a set of SFs (Ψ^n) that we shall define shortly, and the constraint requires that all the policies in Π^n achieve value better than a parameter $\alpha \in [0,1]$ times the value of the optimal policy (here v_e^π is the value of policy π for extrinsic reward r_e and v_e^* is the value of the optimal policy with respect to r_e). Note that α controls how big a space of policies we search over for our diverse set of policies. In general, the smaller the α parameter the larger the set of α-optimal policies and thus the greater the diversity of the policies found in Π^n[1].

Algorithm 1 Diverse Successive Policies

1: **Input:** mechanism to compute rewards r_e and r_d.
2: **Initialize:** $\pi^0 \leftarrow \arg\max_{\pi\in\Pi} r_e \cdot d_\pi$,
3: $v_e^* = v^{\pi^0}$, $\Pi^0 = \{\pi^0\}$
4: **for** $i = 1, \dots, T$ **do**
5: Compute diversity reward $r_d^i = \text{D}(\Psi^{i-1})$
6: $\pi^i = \arg\max_\pi d_\pi \cdot r_d^i$ s.t. $d_\pi \cdot r_e \ge \alpha v_e^*$
7: Estimate the SFs ψ^i of the policy π^i
8: $\Pi^i = \Pi^{i-1} \cup \{\pi^i\}$, $\Psi^i = \Psi^{i-1} \cup \{\psi^i\}$
9: **end for**
10: **return** Π^T

Common to many approaches is to define a diversity objective using *intrinsic* rewards (Gregor et al., 2017; Eysenbach et al., 2019; Kumar et al., 2020; Zahavy et al., 2021), *i.e.*, rewards not from the environment but defined by the agent itself. Our approach also uses intrinsic rewards to induce diversity, as we describe in Algorithm 1. The algorithm receives as input two reward functions r_e and r_d, which together define a CMDP. The reward r_d corresponds to a diversity intrinsic reward. We will discuss five different candidate r_d's. The constraint reward r_e will typically be the extrinsic reward, but we will also consider two alternative choices for r_e. In the initialization step of Algorithm 1 (line 2) there are no policies in the set, and so the goal of the first policy π^0 is to solve the MDP with reward r_e. Algorithm 1 then adds π^0 and its SFs to the set, and the variable v_e^* is set to be v^0. v_e^* defines the near-optimality constraint αv_e^* for the other policies (say with $\alpha = 0.9$).

After this first step, the algorithm proceeds in iterations. In iteration i, an intrinsic reward r_d^i is computed given the pre-

[1]When the extrinsic reward is positive ($r_e(s,a) \ge 0, \forall s,a$), the extrinsic value is positive $v_e^\pi \ge 0, \forall \pi$, and setting $\alpha = 0$ in Eq. (1) is equivalent to the no-reward setting where the goal is to maximize diversity.

vious policies in the set Π^{i-1}. The next policy to be added to the set, π^i, is the solution to the following Constrained MDP (CMDP) (line 6 in Algorithm 1):

$$\arg\max_{\pi} \;\; d_\pi \cdot r_d^i \quad \text{s.t.} \quad d_\pi \cdot r_e \geq \alpha v_e^*. \tag{2}$$

In words, the new policy optimizes the average intrinsic reward value subject to the constraint that it be near-optimal with respect to its average extrinsic reward value. In Section 5 we discuss the details of how to solve Eq. (2). Clearly, the behavior of Algorithm 1 strongly depends on the choice of r_d, the intrinsic reward used to induce diversity. We now discuss five alternatives to define this reward.

4. Measuring Policy Diversity

A key aspect of our method is the measure of diversity. Our focus is on diverse *policies*, as measured by their stationary distribution after they have mixed. This suggests we should measure diversity in the space of SFs, as they are defined under the policy's stationary distribution (see Section 2). In contrast, prior work have focused on learning diverse *skills*, which is often measured before the skill policy mixes. A common approach to measuring skill diversity is to measure skill discrimination in terms of trajectory-specific quantities such as terminal states (Gregor et al., 2017), a mixture of the initial and terminal states (Baumli et al., 2020), or trajectories (Eysenbach et al., 2019). An alternative approach that implicitly induces diversity is to learn policies that maximize the robustness of the set Π^n to the worst-possible reward (Kumar et al., 2020; Zahavy et al., 2021).

In Subsections 4.1 and 4.2, we analyze the diversity of these two approaches in the space of SFs and find that they both depend on the initialization of the algorithm and cannot guarantee diversity. Motivated by these findings, we develop two new explicit diversity rewards that aim to minimize the correlation between the SFs of the policies in the set. We discuss these new methods in Section 4.3.

4.1. Diversity via Discrimination

Discriminative approaches rely on the intuition that skills should be distinguishable from one another simply by observing the states that they visit. Learning diverse skills is then a matter of learning skills that can be easily discriminated. For instance, DIAYN (Eysenbach et al., 2019) maximizes the mutual information between skills and states as follows. Given a probability space $(\Omega, \mathcal{F}, \mathcal{P})$, we denote by $I(S;Z)$ the mutual information between the random variable state $S : \Omega \to \mathcal{S}$ and latent random variable (skill) $Z : \Omega \to \mathcal{Z}$ (Cover, 1999). We also use $H[A|S]$ to refer to the conditional entropy of the action random variable $A : \Omega \to \mathcal{A}$ conditioned on state S. Finally, the conditional mutual information between A and Z given S is denoted by $I(A;Z|S)$. Then, the DIAYN objective to be maximized, given a prior over the latents, p, is:

$$\begin{aligned} &I(S;Z) + H[A|S] - I(A;Z|S) \\ &\quad = H[A|S,Z] + \mathbb{E}_{\substack{z\sim p(z) \\ s\sim d_{\pi^z}}}[\log p(z|s) - \log p(z)]. \end{aligned} \tag{3}$$

This is an entropy-regularized objective that seeks to maximize the information that states contain about the skill used to reach it. In particular, the term of interest is $\mathbb{E}_{z\sim p(z), s\sim d_{\pi^z}}[\log p(z|s) - \log p(z)]$, which corresponds to the value of a skill in an MDP with reward $r(s|z) = \log p(z|s) - \log p(z)$. A skill policy $\pi(a|s,z)$ controls the first component of this reward, $p(z|s)$, which measures the probability of identifying the skill in state s. Hence, the policy is rewarded for visiting states that differentiates it from other skills, thereby implicitly encouraging diversity.

The exact form of $p(z|s)$ depends on how skills are encoded (Gregor et al., 2017). The most common version is to encode z as a one-hot d-dimensional variable (e.g.; Gregor et al., 2017; Achiam et al., 2018; Eysenbach et al., 2019). Similarly, we represent z as $z \in \{1, \dots, n\}$ to index n separate policies π^z. In addition, the concept of finding a small set of meaningful policies is appealing from the interpretability perspective.

$p(z|s)$ is typically intractable to compute due to the large state space and is instead approximated via a learned discriminator $q_\phi(z|s)$. In our case, we measure $p(z|s)$ under the stationary distribution of the policy; that is, $p(s|z) = d_{\pi^z}(s)$. Therefore, for the purpose of analysis, we can find an analytic form for the objective of DIAYN before we apply the variational approximation. Given this, applying Bayes rule to $p(z|s)$ yields

$$p(z|s) = \frac{d_{\pi^z}(s)p(z)}{\sum_k d_{\pi^k}(s)p(k)}. \tag{4}$$

And in the kernel case, we define a Gibbs distribution

$$p(z|s) = \frac{p(z)\exp\left(\phi(s)\cdot\psi^z\right)}{\sum p(k)\exp(\phi(s)\cdot\psi^k)}. \tag{5}$$

Plugging $p(z|s)$ from Eq. (4) in the objective of DIAYN, the relevant term in Eq. (3) becomes

$$\begin{aligned} &\mathbb{E}_{z\sim p(z), s\sim d(\pi^z)}[\log p(z|s)] \\ &\quad = \sum_z p(z) \sum_s d_{\pi^z}(s) \log\left(\frac{d_{\pi^z}(s)p(z)}{\sum_k d_{\pi^k}(s)p(k)}\right). \end{aligned} \tag{6}$$

Finding a policy with maximal value for this reward can be seen as solving an optimization program in d_{π^z} under the constraint that the solution is a valid stationary state distribution (Section 2). The term $\sum_s p(s|z)\log p(s|z)$ corresponds to the negative entropy of d_{π^z}, meaning that the objective to be maximized is *convex* in d_{π^z}.

Lemma 1. *The function* $\sum_s d_{\pi^z}(s)\log\left(\frac{d_{\pi^z}(s)p(z)}{\sum_k d_{\pi^k}(s)p(k)}\right)$ *is a convex function of* d_{π^z}.

The proof can be found in Appendix A; briefly, Lemma 1 holds because the function can be written as $KL(d_{\pi^z}||\sum_k p(k)d_{\pi^k}) + \sum_s d_{\pi^z}(s)\log p(z)$ and the KL-divergence is jointly convex on both arguments (Boyd and Vandenberghe, 2004, Example 3.19). The convexity of the objective results from the fact that the intrinsic reward $\log p(z|s)$ is a function of the policy. In the standard RL setup, the reward is not a function of the policy and the objective is linear in it, thus, maximizing and minimizing the reward are both convex minimization problems. However, when the reward is a function of the policy, maximization and minimization of the reward are not equivalent optimization problems. In DIAYN, the maximization of $\log p(z|s)$ leads to convex maximization while the minimization of the same reward leads to convex minimization. We note that the convexity of the objective has nothing to do with the variational approximation typically used to compute $p(z|s)$; it is encountered with or without it.

The observation that discriminatory objectives lead to a set of n convex maximization problems in our setting is problematic, since the optimality of the solutions—in particular, their diversity—cannot be guaranteed. From the perspective of the policy set, the algorithm may converge to a set which is a local maxima rather than the global maxima, and therefore result in suboptimal diversity. In practice, different initializations and stochastic updates might mitigate the issue to some degree. In addition, it is possible that all the local maxima are close to optimal. For example, similar observations were made regarding the loss surface of deep neural networks, but the local optima points were shown to be very good in practice (Dauphin et al., 2014; Choromanska et al., 2015; Soudry and Carmon, 2016), mitigating the issues mentioned above. Thus, we recommend taking Lemma 1 as an observation regarding the optimization landscape of DIAYN which we hope to further explore in future work.

4.2. Diversity via Robustness

An alternative approach that implicitly induces diversity is to seek robustness among a set of policies by maximizing the performance w.r.t the worst case reward (Kumar et al., 2020; Zahavy et al., 2021); for fixed n, the goal is:

$$\max_{\Pi^n \subseteq \Pi} \min_{w\in B_2} \max_{\pi^i \in \Pi^n} \psi^i \cdot w. \tag{7}$$

Here B_2 is the ℓ_2 unit ball, Π is the set of all possible policies, $\Pi^n = \{\pi^1, \dots, \pi^n\}$ is the set of n policies for which we are optimizing. Let us parse this objective term by term. First, the inner product $\psi^i \cdot w$ yields the expected value under the steady-state distribution (see Section 2) of the policy π^i. The inner min-max is a two-player zero-sum game, where the minimizing player is finding the *worst-case* reward function (since weights and reward functions are in a one-to-one correspondence) that minimizes the expected value, and the maximizing player is finding the best policy from the set Π^n (since policies and SFs are in a one-to-one correspondence) to maximize the value. The outer maximization is to find the best set of n policies that the maximizing player can use.

Intuitively speaking, the solution Π^n to this problem might be a diverse set of policies since a non-diverse set is likely to yield a low value of the game, that is, it would easily be exploited by the minimizing player. In this way diversity and robustness are dual to each other, in the same way as a diverse financial portfolio is more robust to risk than a heavily concentrated one. By forcing our policy set to be robust to an adversarially chosen reward it will be diverse.

In (Kumar et al., 2020), the authors proposed a solution to Eq. (7) using a CMDP with r_d as discrimination (via DIAYN) and r_e is the extrinsic reward; we discuss it in more detail in Section 5. In (Zahavy et al., 2021), the authors proposed an iterative solution to Eq. (7) that incrementally adds policies to a solution set Π^n (Algorithm 2 in the appendix). The authors define a Set Max Policy (SMP) as a policy that takes a set of policies and a reward as inputs and returns the best policy in the set for this reward. In each iteration, the algorithm computes the worst case reward w.r.t to the SMP, finds the policy that maximizes it, and adds it to the set. In iteration n The value of the SMP on the set Π^n is defined as $v^n = \min_{w\in B_2} \max_{\pi^i\in\Pi^n} \psi^i \cdot w$, and it is guaranteed that this value strictly increases $v^{n+1} > v^n$ in each iteration until the optimal solution is found. The following Lemma suggests that this procedure is equivalent to a fully corrective FW (Frank and Wolfe, 1956) algorithm on the function $f = ||\cdot||_2$. As a consequence, it is guaranteed to convergence to the optimal solution in a linear rate (Jaggi and Lacoste-Julien, 2015).

Lemma 2. *The iterative procedure in (Zahavy et al., 2021) is equivalent to a fully corrective FW algorithm to minimize the function $f = ||\psi^\pi||_2$. As a consequence, to achieve an $\epsilon-$optimal solution, the algorithm requires at most $O(\log(1/\epsilon))$ iterations.*

The proof in Appendix B suggests that the SMP policy is equivalent to the fully corrective search (maintaining a dictionary of solutions from previous iterations and choosing the best convex combination). The only difference between the two algorithms is that one of them solves a max-min problem where the other solves the equivalent min-max problem, and therefore they are guaranteed to have the same iterations from strong duality. Unfortunately this approach, like the discriminative approaches, has a weakness that can limit the ultimate diversity in the set. To see this note that

$$\begin{aligned}
&\max_{\Pi^n \subseteq \Pi} \min_{w\in B_2} \max_{\pi^i \in \Pi^n} \psi^i \cdot w \le \min_{w\in B_2} \max_{\pi^i \in \Pi} \psi^i \cdot w \\
&= \max_{\pi^i \in \Pi} \min_{w\in B_2} \psi^i \cdot w = -\min_{\pi^i \in \Pi} \|\psi^i\| \stackrel{\text{def}}{=} v^*,
\end{aligned}$$

where the inequality comes from the fact that $\Pi^n \subseteq \Pi$, and the first equality uses von Neumann's minimax theorem (von Neumann, 1928). If we let $\pi^* = \arg\min_{\pi^i\in\Pi} \|\psi^i\|$, then if $\Pi^n = \{\pi^*\}$ we have an optimal policy set for the game, since we have found a policy set that achieves the known upper bound on the value of the game, v^*. In other words a single policy is a sufficient solution for Eq. (7), which is problematic since the goal was to build up a set of many diverse policies. Similar to the discriminative approaches, in practice we obtain more policies by initializing the set away from π^*, or alternatively restricting Π^n to deterministic policies. However, this issue likely explains the empirical observations in (Zahavy et al., 2021) that there are only a few active policies in the optimal sets.

Note that the results above hold only in the case that Π is the set of all the stochastic policies in the MDP; if only deterministic policies are used, we cannot apply the von Neumann's minimax theorem. This is not an issue since we are interested in stochastic policies for multiple reasons: optimal solutions to CMDPs are stochastic policies (Altman, 1999) and stochastic policies are the most common approach in continuous control tasks, which is the focus of our experiments.

4.3. Explicit diversity methods

The two diversity mechanisms we have discussed so far were designed to maximize robustness or discrimination. Each one has its own merits in terms of diversity, but since they do not explicitly maximize a diversity measure they cannot guarantee that the resulting set of policies will be diverse. We now propose two reward signals designed to induce a diverse set of policies. The way they do so is to leverage the information about the policies' long-term behavior available in their SFs. Both rewards are based on the intuition that the correlation between SFs should be minimized.

To motivate this approach, we note that SFs can be seen as a compact representation of a policy's stationary distribution. This becomes clear when we consider the case of a finite MDP with $|\mathcal{S}|$-dimensional "one-hot" feature vectors ϕ whose elements encode the states: $\phi_i(s) = \mathbb{I}\{s = i\}$, where $\mathbb{I}\{\cdot\}$ is the indicator function. In this special case the SFs of a policy π coincide with its stationary distribution, that is, $\psi^\pi = d_\pi$. Under this interpretation, minimizing the correlation between SFs intuitively corresponds to encouraging the associated policies to visit different regions of the state space—which in turn leads to diverse behavior. As long as we assume the tasks of interest are linear combinations of the features $\phi \in \mathbb{R}^d$, which we do, similar reasoning applies when $d < |\mathcal{S}|$.

But how do we compute policies in order to minimize the correlation between their SFs? To answer this question, we first consider the extreme scenario where there is a single policy π^k in the set Π. In this case the objective is: $\max_{\psi^z} \psi^z \cdot w$, where $w = -\psi^k$. Solving this problem is an RL problem whose reward is linear in the features weighted by w. A similar objective was investigated in (Hansen et al., 2020), but there w was sampled i.i.d from a fixed prior. The question we are trying to address is: how to define w taking into account multiple policies in the set Π^n?

We propose two answers to this question. The first one is to have w be the negative average of the SFs of the policies currently in the set, that is, $w = -\frac{1}{k}\sum \psi^k$. This formulation is useful as it measures the sum of negative correlations within the set. However, when two policies in the set happen to have the same SFs with opposite signs, they cancel each other, and do not impact the diversity measure. This diversity objective shares some similarities with the novelty search algorithm in (Conti et al., 2018), where the mean pairwise distance between the current policy and an archive of other policies is used.

The second diversity-inducing reward we propose addresses this issue. It is defined as the minimum over the SFs in each state: $r(s) = \min_k \left\{\phi(s) \cdot -\psi^k\right\}$. This objective encourages the policy to have the largest "margin" from the set, as it maximizes the negative correlation from the element that is "closest" to it. This objectives shares some similarities with a recent work (Parker-Holder et al., 2020) that uses the determinant of the kernel matrix and penalizes it to the closest agents in the population, building on ideas from Determinantal point processes (Kulesza et al., 2012). Finally, we note that we also apply a non linear transformation to bound both of these rewards; the details are in the supplementary (Appendix C).

5. Solving the constrained MDP

At the core of our approach is the solution of a CMDP. The literature on CMDPs is quite vast and we refer the reader to (Altman, 1999) and (Szepesvári, 2020) for treatments of the subject at different levels of abstraction. In this work we will focus on a reduction of CMDPs to MDPs via gradient updates. The idea is to look at the Lagrangian of Eq. (2):

$$L(\pi, \lambda) = -d_\pi \cdot (r_d + \lambda r_e) - \lambda \alpha v_e^*. \tag{8}$$

Then, solving the CMDP in Eq. (2) is equivalent to solving $\min_{\pi\in\Pi} \max_{\lambda\geq 0} L(\pi, \lambda)$.

Solving CMDPs via Lagrangian methods dates back to (Borkar, 2005; Bhatnagar and Lakshmanan, 2012); more recently the problem has been tackled using Deep RL techniques (Tessler et al., 2019; Calian et al., 2021). These algorithms perform primal-dual gradient updates on the min-max game. When the value function of the policy satisfies the constraint, the Lagrange multiplier will decrease, putting more emphasis on the extrinsic reward; when the constraint is not satisfied, the Lagrange multiplier will increase to satisfy the constraint.

Non linear Lagrange multiplier. We would like our agent to optimize a bounded reward signal, and we discuss how to bound each reward r_d in the supplementary (Appendix C). To guarantee that a combination of two bounded rewards remains bounded, it is sufficient to combine them via a convex combination. To achieve that, we use a Sigmoid activation on the Lagrange multiplier so the reward is a convex combination of the diversity and the extrinsic rewards:

$$r(s) = \sigma(\lambda) r_e(s) + (1 - \sigma(\lambda)) r_d(s).$$

We further introduce an entropy regularization on λ to prevent $\sigma(\lambda)$ from getting to extreme values (1 or 0), where the

Sigmoid activation is saturated and has low gradients. This can happen, for example, at the beginning of learning where the agent's policy is sub-optimal and does not satisfy the constraint for many iterations. The objective for λ is thus:

$$f(\lambda) = \sigma(\lambda)(\hat{v} - \alpha v_e^*) - a_h H(\sigma(\lambda)), \tag{9}$$

where H is the entropy function, a_h is the weight of the entropy regularization and $\hat{v}$ is an estimate of the total cumulative extrinsic return that the agent obtained in recent trajectories. The Lagrangian λ is updated by performing gradient descent on Eq. (9) every N_λ agent steps.

Estimation of average rewards. Another important step of Algorithm 1 which is not directly related to solving the CMDP is the estimation of the average rewards. For that, we used a simple Monte Carlo estimates: $\tilde{v}_j = \frac{1}{T}\sum_{t=1}^T r_t$, i.e, the empirical average reward obtained by the agent in trajectory j (where $T = 1000$). We used the same estimator to estimate the average SFs (replace r_t with ϕ_t).

The value $\tilde{v}_j$ is a good estimate of the average reward, but it is not perfect. The issue is that the trajectory is of finite length, and therefore the samples in the beginning of the trajectory, before the policy is mixed, are biased. Our experiments are in the DM control suite (Tassa et al., 2018) where the mixing time is small; the policies we discover roughly mix after ~ 50 steps (as can be seen in the videos in the supplementary). Since the mixing time is much shorter than T, the effect of the biased samples is small ($\sim 5\%$). It is also possible to wait until the policy is mixed or to collect a perfect unbiased estimate of the average reward via Coupling From the Past procedure (Propp and Wilson, 1996) as was done in (Zahavy et al., 2020a). Note that this is a known issue with any practical policy gradient method but was not found to make a big difference empirically.

We further average the estimate using a running average with decay factor of a_d: $\hat{v}_j = a_d\hat{v}_{j-1} + (1 - a_d)\tilde{v}_j$; this is the estimate we use in Eq. (9). The running average variables are set to 0 between iterations of Algorithm 1. Finally, we note here that we also experimented with the discounted criteria (discounted SFs). In that case, we observed that there is too much emphasis on the features that are observed at the beginning of the trajectory, resulting in less diversity across the entire trajectory.

Discussion. A different feasible approach to combine r_d and r_e is to model the problem as a *multi-objective* MDP. That is, the diversity objective is added to the main one via a fixed, stationary weighting of the two rewards, e.g., $r = a_1 r_d + a_2 r_e$. We note that the solution of such a multi-objective MDP cannot be be a solution to a CMDP. I.e., it is not possible to find the optimal dual variables λ^*, plug them in Eq. (8) and simply solve the resulting (unconstrained) MDP. Such an approach ignores the fact the dual variables must be a 'best-response' to the policy and is referred to as the "scalarization fallacy" in (Szepesvári, 2020, Section 4).

While Multi objective MDPs have been used in prior QD-RL papers (Hong et al., 2018; Masood and Doshi-Velez, 2019; Parker-Holder et al., 2020; Gangwani et al., 2020; Peng et al., 2020; Zhang et al., 2019), we now outline a few potential advantages for using CMDPs. First, the CMDP formulation guarantees that the policies that we find are near optimal (satisfy the constraint). Secondly, the weighting coefficient in multi-objective MDPs has to be tuned, while in our case it is being adapted over time. This is particularly important in the context of maximizing diversity while satisficing reward. In many cases, as we observed in our experiments, the diversity reward might have no other option other than being the negative of the extrinsic reward. In these cases our algorithm will return good policies that are not diverse, while a solution to multi-objective MDP might fluctuate between the two objectives and not be useful at all.

CMDPs in related QD papers. Kumar et al. (Kumar et al., 2020) proposed that solving a CMDP with r_d as discrimination reward and r_e as the extrinsic reward will lead to a solution to the robustness objective (Eq. (7)). Sun et al. (Sun et al., 2020) also investigated CMDPs, but focused on the setup where the diversity reward has to satisfy a constraint, so the diversity reward is r_e and the extrinsic reward is r_d. But most importantly, we use a different method to solve CMDPs, which is based on Lagrange multipliers and SFs and is justified from CMDP theory (Altman, 1999; Borkar, 2005; Bhatnagar and Lakshmanan, 2012), while these other two papers use techniques that are not guaranteed to solve CMDPs.

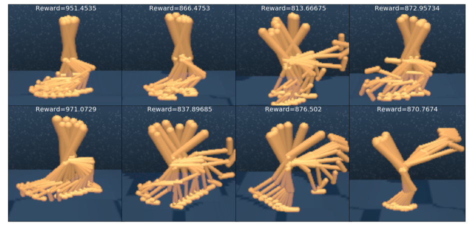
6. Experiments

We conducted our experiments on domains from the DM Control Suite (Tassa et al., 2018), standard continuous control locomotion tasks where diverse near-optimal policies should naturally correspond to different gaits. We focused on the setup where the agent is learning from feature observations corresponding to the positions and velocities of the body joints being controlled by the agent. Due to space considerations, we focus on domains where the diversity is interesting from a visual point of view, and in particular on Walker and Dog. In simpler domains like Cartpole and Reacher, we observed simple symmetric diversity – one policy moves a certain way clockwise and then the second policy moves in the same way anti-clockwise (see Fig. 5 in the supplementary). Later policies in the set are less distinguishable visually but can learn, for example, to balance the pole while moving. Note that without a diversity mechanism, the agent tends to only move in a single direction (e.g. clockwise).

In most of our experiments, the extrinsic reward r_e, which defines the optimality constraint in Algorithm 1, is set to be the environment reward provided by the DM Control Suite . The first policy in the set is trained to only maximize the extrinsic reward, and the other policies has to satisfy the constraint of being $\alpha = 0.9$ optimal w.r.t it. In these experiments, we report the reward that each policy collects in white color on top of each figure. Additionally we report

#	r_e	%
1	920	100
2	809	88
3	820	89
4	878	95
5	818	89
6	818	89
7	490	53
8	926	101

(a) Walker Stand, r_e as reward; r_d as min.



#	r_e	%
1	951	100
2	866	91
3	813	85
4	872	92
5	971	102
6	837	88
7	876	92
8	870	91

(b) Walker Walk, r_e as reward; r_d as average.

Figure 1: Diverse near optimal policies in Walker

the reward of each policy in a small table in the main text.

In the QD community, there is no consensus regarding a single metric for measuring diversity, and some argue that there shouldn't be such (see, for example, the book "Why Greatness Cannot Be Planned" (Stanley and Lehman, 2015)). Inspired by this literature, we focus on measuring diversity only qualitatively by visualizing the learned policies. We strongly recommend the reader to check our visualization website where we show **videos** of the trajectories that each policy takes at https://anon98723.github.io/. In addition, we present "motion figures" by discretizing the videos (details in the Appendix) that give a fair impression of the policy behaviours. We would like to note that we did not tune our method to maximize diversity based on any metric other than constraint satisfaction (maintaining near-optimality). The main purpose of our experiments are the feasibility of the CMDP framework as proposed in Algorithm 1, i.e., to demonstrate that we discover diverse near-optimal policies.

Choice of r_d: Given that our Diverse Successive Policies algorithm (1) can be used with different measures of diversity, we compared four different choices. The previously proposed robustness and discrimination measures and the new min and average explicit measures of diversity we proposed in Section 4.3, corresponding to: **(1) Robustness:** the worst case linear reward with respect to the previous policies in the set: $r_d(s) = w \cdot \phi(s)$, where $w = \min_{w \in B_2} \max_{z \in [1,..,n-1]} \psi^z \cdot w$ is the internal minimization in Eq. (7). **(2) Discrimination** $r_d(s) = \log(\frac{\exp{(\phi(s) \cdot \psi^n)}}{\sum_{i=j}^n \exp(\phi(s) \cdot \psi^j)})$, where ψ^n is the running average estimator of the SFs of the current policy. This reward corresponds to Eq. (5) with a uniform prior. **(3) Min:** $r_d(s) = \min_{z \in [1,..,n-1]} -\psi^z \cdot \phi(s)$. **(4) Average:** $r_d(s) = -\frac{1}{n-1} \sum_{j=1}^{n-1} \psi^j \cdot \phi(s)$. **(5) None:** $r_d(s) = 0$ or no diversity.

Fig. 1a presents eight polices that were discovered by Algorithm 1 where r_d is the minimum explicit diversity criteria for **Walker.stand**. As we can see, the policies exhibit different types of standing: standing on both legs, standing on either leg, lifting the other leg forward and backward, spreading the legs and stamping. Not only are the policies different from each other, they also achieve high extrinsic reward in standing (see values on top of each policy visualization). Similar figures for the other diversity mechanisms can be found in the supplementary material (Appendix D.2). We observed that in this domain the Average diversity criterion can also discover policies that behave differently, but they are not as diverse as the ones found using the Minimum criterion (see Appendix D.2 in the supplementary material)

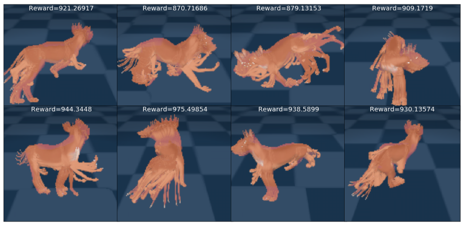
The robustness mechanism can also provide diverse policies, but it tends to converge after a few iterations so no further diversity is achieved by the algorithm after 3 iterations. We also include a figure of different policies with no diversity mechanism in the supplementary (Fig. 10); in this case there is a small amount of diversity from training, but it is much less significant than the diversity we get with a diversity objective. Similarly, the discrimination method exhibits diversity but not as good as the explicit methods. We believe that this is due to the fact that the policies that maximize the extrinsic reward are already discriminative, and the algorithm fails to escape these local minima.

Fig. 1b presents similar results in the **Walker.walk** environment where r_d is the average explicit diversity criteria. In this case the walker discovered how to walk in different ways, such as lifting one of the legs while up walking, walking with high knees, or walking with the heels to the bottom. In this domain we observed much better diversity with the explicit diversity mechanisms than with robustness or discrimination, see Appendix D.3. We also note that in both of the Walker environments, all (but one) of the discovered policies that we found are indeed near optimal, and satisfy the constraint (which was set to 90%).

Fig. 2 presents results in the **Dog.stand** environment where in Fig. 2a r_d is the minimum explicit diversity criteria and in Fig. 2b there is no diversity mechanism. Inspecting Fig. 2b we can see that the dog learns how to stand (different policies are independent of each other so we leave the % blank), but in all cases, it stands with four legs on the ground. On the other hand, in Fig. 2a the dog learns different variations of "three leg standing" (lifting one of his legs), and still achieves high reward.

Next, we present results in the **no-reward** setting, where the agent has no access to the reward from the environment. Our results with None diversity confirm that the implementation of these diversity mechanisms yields complex locomotion in the no-reward setting as was reported in the original papers. However, in more complex domains like Walker,

#	r_e	%
1	921	100
2	870	94
3	879	95
4	909	98
5	944	102
6	975	106
7	938	102
8	930	101



(a) Dog Stand, r_e as reward; r_d as min.

#	r_e	%
1	812	—
2	936	—
3	765	—
4	892	—
5	926	—
6	891	—
7	921	—
8	948	—

(b) Dog Stand, r_e as reward; r_d as none.

Figure 2: Diverse near optimal policies in Dog

without adding the explicit diversity we get static behaviours that resemble "Yoga" exercises, as was also reported, for example, in (Zahavy et al., 2021).

Fig. 3 presents results for **Walker** where r_e is robustness and r_d is average. Inspecting the results, we can see that the agent discovered complex locomotion skills such as kneeling backwards, crawling and flick-flack jumping. We also report the extrinsic reward for standing as another measure of zero-shot transfer (it was not used during training at all). In Appendix D.4 we can see that other diversity mechanisms discovered other surprising skills such as "head walking".

Figure 3: Walker, r_e as robustness and r_d as average.

Finally, Fig. 4 presents results for **Cheetah** where r_e is discrimination and r_d is robustness. The cheetah learns to run forward, backwards, and then to do various jumps. While previous methods were able to discover similar behaviours, they are typically not that diverse with such a small set.

Figure 4: Cheetah, r_e as discrimination and r_d as robustness.

7. Conclusion

In this work we proposed a framework for discovering near optimal diverse behaviours. We framed the problem as solving a CMDP where a diversity intrinsic reward and the extrinsic reward are adaptively combined. There are interesting connections to whitebox metagradients (Xu et al., 2018; Zahavy et al., 2020c) – the updates of the Lagrangian can be viewed as the outer update in metagradients where satisfying the constraint is the outer loss. Using metagradients to learn other diversity hyperparameters or even to discover the diversity reward itself (Zheng et al., 2018) are exciting directions for future work. Key to our approach was the idea of measuring diversity in the space of SFs. This design choice allowed us to provide insights on how existing diversity mechanisms behave from the perspective of convex optimization.

There are many exciting applications for our framework. For example, consider the process of using RL to train a robot to walk. The designer does not know *a priori* which reward will result in the desired walking pattern. Thus, robotic engineers often train a policy to maximize an initial reward, tweak the reward, and iterate until they reach the desired behaviour. Using our approach, the engineer would have multiple forms of walking to choose from in each attempt, which are also interpretable (linear in the weights).

References

J. Achiam, H. Edwards, D. Amodei, and P. Abbeel. Variational option discovery algorithms. *arXiv preprint arXiv:1807.10299*, 2018.

A. Agarwal, M. Henaff, S. Kakade, and W. Sun. Pc-pg: Policy cover directed exploration for provable policy gradient learning. *arXiv preprint arXiv:2007.08459*, 2020.

E. Altman. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

A. Barreto, W. Dabney, R. Munos, J. J. Hunt, T. Schaul, H. P. van Hasselt, and D. Silver. Successor features for transfer in reinforcement learning. In *Advances in neural information processing systems*, pages 4055–4065, 2017.

K. Baumli, D. Warde-Farley, S. Hansen, and V. Mnih. Relative variational intrinsic control. *arXiv preprint arXiv:2012.07827*, 2020.

R. Behovits. Game 8: Leko wins to take the lead, 2004. URL https://en.chessbase.com/post/game-8-leko-wins-to-take-the-lead.

S. Bhatnagar and K. Lakshmanan. An online actor–critic algorithm with function approximation for constrained markov decision processes. *Journal of Optimization Theory and Applications*, 153(3):688–708, 2012.

V. S. Borkar. An actor-critic algorithm for constrained markov decision processes. *Systems & control letters*, 54 (3):207–213, 2005.

S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

D. A. Calian, D. J. Mankowitz, T. Zahavy, Z. Xu, J. Oh, N. Levine, and T. Mann. Balancing constraints and rewards with meta-gradient d4pg. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=TQt98Ya7UMP.

A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun. The loss surfaces of multilayer networks. In *Artificial intelligence and statistics*, pages 192–204. PMLR, 2015.

E. Conti, V. Madhavan, F. P. Such, J. Lehman, K. O. Stanley, and J. Clune. Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 5032–5043, 2018.

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

C. da Fonseca-Wollheim. Swapping songs with chess grandmaster garry kasparov, 2020. URL https://www.nytimes.com/2020/12/18/arts/music/garry-kasparov-classical-music.html.

Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper/2014/file/17e23e50bedc63b4095e3d8204ce063b-Paper.pdf.

B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=SJx63jRqFm.

M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

T. Gangwani, J. Peng, and Y. Zhou. Harnessing distribution ratio estimators for learning agents with quality and diversity. *arXiv preprint arXiv:2011.02614*, 2020.

K. Gregor, D. J. Rezende, and D. Wierstra. Variational intrinsic control. *International Conference on Learning Representations, Workshop Track*, 2017. URL https://openreview.net/forum?id=Skc-Fo4Yg.

S. Hansen, W. Dabney, A. Barreto, D. Warde-Farley, T. V. de Wiele, and V. Mnih. Fast task inference with variational intrinsic successor features. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJeAHkrYDS.

Z.-W. Hong, T.-Y. Shann, S.-Y. Su, Y.-H. Chang, T.-J. Fu, and C.-Y. Lee. Diversity-driven exploration strategy for deep reinforcement learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 10510–10521, 2018.

M. Jaggi and S. Lacoste-Julien. On the global linear convergence of frank-wolfe optimization variants. *Advances in Neural Information Processing Systems*, 28, 2015.

A. Kulesza, B. Taskar, et al. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.

S. Kumar, A. Kumar, S. Levine, and C. Finn. One solution is not all you need: Few-shot extrapolation via structured maxent rl. *Advances in Neural Information Processing Systems*, 33, 2020.

M. A. Masood and F. Doshi-Velez. Diversity-inducing policy gradient: Using maximum mean discrepancy to find a set of diverse policies. *arXiv preprint arXiv:1906.00088*, 2019.

B. Matusch, J. Ba, and D. Hafner. Evaluating agents without rewards. *arXiv preprint arXiv:2012.11538*, 2020.

N. Mehta, S. Natarajan, P. Tadepalli, and A. Fern. Transfer in variable-reward hierarchical reinforcement learning. *Machine Learning*, 73(3):289, 2008.

J.-B. Mouret and J. Clune. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*, 2015.

A. F. Osborn. Applied imagination. 1953.

J. Parker-Holder, A. Pacchiano, K. M. Choromanski, and S. J. Roberts. Effective diversity in population based reinforcement learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Z. Peng, H. Sun, and B. Zhou. Non-local policy optimization via diversity-regularized collaborative exploration. *arXiv preprint arXiv:2006.07781*, 2020.

J. G. Propp and D. B. Wilson. Exact sampling with coupled markov chains and applications to statistical mechanics. *Random Structures and Algorithms*, 1996.

J. K. Pugh, L. B. Soros, and K. O. Stanley. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3:40, 2016.

M. L. Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 1984.

S. Singh, R. L. Lewis, A. G. Barto, and J. Sorg. Intrinsically motivated reinforcement learning: An evolutionary perspective. *IEEE Transactions on Autonomous Mental Development*, 2(2):70–82, 2010.

D. Soudry and Y. Carmon. No bad local minima: Data independent training error guarantees for multilayer neural networks. *arXiv preprint arXiv:1605.08361*, 2016.

K. O. Stanley and J. Lehman. *Why greatness cannot be planned: The myth of the objective*. Springer, 2015.

H. Sun, Z. Peng, B. Dai, J. Guo, D. Lin, and B. Zhou. Novel policy seeking with constrained optimization. *arXiv preprint arXiv:2005.10696*, 2020.

C. Szepesvári. Constrained mdps and the reward hypothesis, 2020. URL https://readingsml.blogspot.com/2020/03/constrained-mdps-and-reward-hypothesis.html.

Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. d. L. Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

C. Tessler, D. J. Mankowitz, and S. Mannor. Reward constrained policy optimization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=SkfrvsA9FX.

J. von Neumann. Zur theorie der gesellschaftsspiele. *Mathematische annalen*, 100(1):295–320, 1928.

Z. Xu, H. van Hasselt, and D. Silver. Meta-gradient reinforcement learning. *arXiv preprint arXiv:1805.09801*, 2018.

T. Zahavy, A. Cohen, H. Kaplan, and Y. Mansour. Average reward reinforcement learning with unknown mixing times. *The Conference on Uncertainty in Artificial Intelligence (UAI)*, 2020a.

T. Zahavy, A. Hasidim, H. Kaplan, and Y. Mansour. Planning in hierarchical reinforcement learning: Guarantees for using local policies. In *Algorithmic Learning Theory*, pages 906–934, 2020b.

T. Zahavy, Z. Xu, V. Veeriah, M. Hessel, J. Oh, H. P. van Hasselt, D. Silver, and S. Singh. A self-tuning actor-critic algorithm. *Advances in Neural Information Processing Systems*, 33, 2020c.

T. Zahavy, A. Barreto, D. J. Mankowitz, S. Hou, B. O'Donoghue, I. Kemaev, and S. Singh. Discovering a set of policies for the worst case reward. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PUkhWz65dy5.

Y. Zhang, W. Yu, and G. Turk. Learning novel policies for tasks. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7483–7492. PMLR, 09–15 Jun 2019. URL http://proceedings.mlr.press/v97/zhang19q.html.

Z. Zheng, J. Oh, and S. Singh. On learning intrinsic rewards for policy gradient methods. *arXiv preprint arXiv:1804.06459*, 2018.

A. Proof for Lemma 1

Proof. We will focus on the case that there are no zero elements in d_π which is a standard assumption in ergodic MDPs. Under this assumption f is a twice differentiable function so it is convex if its Hessian is positive semidefinite.

Recall that the prior $p(z)$ is constant, and that the policies $k = 1, ..., k \neq z$ are also constant from the perspective of d_{π^z}. We can therefore introduce a simplified notation and write the objective as

$$\sum_s d_{\pi^z}(s) \log \left(\frac{d_{\pi^z}(s)p}{d_{\pi^z}(s)p + c_s} \right)$$

The variable d_{π^z} is a vector in the $|S|-$simplex. We can represent it using $|S| - 1$ degrees of freedom $x_1, ..x_{s-1} \in [0, 1]$ where the last element is $x_s = 1 - \sum_{i=1}^{|S|-1} x_i$. Notice that x_s is a function of x_i so it has a derivative with respect to. x_i which equals -1. So we have

$$f(x) = \sum_i x_i \log \left(\frac{x_i p}{x_i p + c_i} \right) + x_s \log \left(\frac{x_s p}{x_s p + c_s} \right)$$

The first derivative of this function with respect to $x_i, i \in [1, .., |S| - 1$ is

$$\begin{aligned}
\frac{\partial f}{\partial x_i} &= \log(x_i) + 1 + \log(p) - \frac{x_i p}{x_i p + ci} - \log(x_i p + c_i) \\
&- \log(x_s) - 1 - \log(p) + \frac{x_s p}{x_s p + c_s} + \log(x_s p + c_s) \\
&= \log(x_i) - \log(x_i p + c_i) - \frac{x_i p}{x_i p + ci} && (10) \\
&- \left(\log(x_s) - \log(x_s p + c_i) - \frac{x_s p}{x_s p + c_s} \right) && (11)
\end{aligned}$$

We can see that the terms in Eq. (10) depend only on x_i and the terms in Eq. (11) depend only on x_s. In addition, we will soon see that the derivatives of x_s will be equal for any $j \in 1, ..., s - 1$. These two observations imply that the Hessian will have the form of

$$H = D + m\underline{1},$$

where D is a diagonal matrix with derivatives of Eq. (10) with respect to x_i as it elements, $\underline{1}$ is a matrix of all ones, and m is the derivative of Eq. (11) with respect to x_j which we will show to be equal for all j. Notice that $\forall x$, we have that $x^T(D + m\underline{1})x = \sum D_i x_i^2 + m(\sum x_i)^2$. This implies that in order for the Hessian to be positive definite, we only need to show that the elements of d and the scalar m are positive. The derivative of Eq. (10) with respect to x_i is

$$\begin{aligned}
&\frac{1}{x_i} - \frac{p}{px_i + c_i} - \frac{p(px_i + ci) - p^2 x_i}{(px_i + c_i)^2} \\
&= \frac{px_i + c_i - px_i}{x_i(px_i + c_i)} - \frac{pc_i}{(px_i + c_i)^2} \\
&= \frac{c_i(px_i + c_i) - px_i c_i}{x_i(px_i + c_i)^2} = \frac{c_i^2}{x_i(px_i + c_i)^2}, && (12)
\end{aligned}$$

which is positive because $x_i \geq 0$.

Similarly, The derivative of Eq. (11) with respect to x_j is

$$\begin{aligned}
&\frac{1}{x_s} - \frac{p}{px_s + c_s} - \frac{p(px_s + cs) - p^2 x_s}{(px_s + c_s)^2} \\
&= \frac{c_s^2}{x_s(px_s + c_s)^2}, && (13)
\end{aligned}$$

which is also positive because $x_s \geq 0$ and concludes our proof. ∎

B. Proof for Lemma 2

Algorithm 2 The iterative procedure in (Zahavy et al., 2021)

Initialize: Sample $\mathbf{w} \sim N(\bar{0}, \bar{1}), \Pi^0 \leftarrow \{\ \}, \pi^1 \leftarrow \arg\max_{\pi\in\Pi} \mathbf{w} \cdot \boldsymbol{\psi}^{\pi}$, $t \leftarrow 1$
$\bar{v}^{\text{SMP}}_{\Pi^1} \leftarrow -||\boldsymbol{\psi}^1||$
repeat
  $\Pi^t \leftarrow \Pi^{t-1} \cup \{\pi^t\}$
  $\Psi^t = \Psi^{t-1} \cup \{\psi^t\}$
  $\bar{\mathbf{w}}^{\text{SMP}}_{\Pi^t} \leftarrow \arg\min_{\mathbf{w}\in\mathbb{B}_2} \max_{\psi\in\Psi^i} w \cdot \psi$
  $\pi^{t+1} \leftarrow \arg\max_{\pi} \psi(\pi) \cdot \bar{\mathbf{w}}^{\text{SMP}}_{\Pi^t}$
  $t \leftarrow t+1$
until $v^t_{\bar{\mathbf{w}}^{\text{SMP}}_{\Pi^n}} \leq \bar{v}^{\text{SMP}}_{\Pi^{t-1}}$
return Π^{t-1}

Algorithm 3 Fully corrective FW for $h(\psi) = 0.5||\psi||_2^2$

Initialize: Let π^1 be a random policy and let ψ^1 be its SFs. Also, let $\Pi^0 = \{\}$ and $\Psi^0 = \{\}$ and $t \leftarrow 1$.
repeat
  $\Pi^t = \Pi^{t-1} \cup \{\pi^t\}$
  $\Psi^t = \Psi^{t-1} \cup \{\psi^t\}$
  $\hat{\psi} = \arg\min_{\psi\in\text{Co}(\Psi^i)} 0.5||\psi||_2^2$.
  $\pi^{t+1} = \arg\max_{\pi} \psi(\pi) \cdot -\nabla h(\hat{\psi}) = \arg\max_{\pi} \psi(\pi) \cdot -\hat{\psi}$
  $t \leftarrow t+1$
until $h(\hat{\psi}) \leq \epsilon$
return Π^{t-1}

In this section we show that the iterates of the fully corrective FW algorithm (Algorithm 3) correspond to the iterates of the Worst Case Policy Iteration algorithm (Algorithm 2). Examining the two algorithms, it is easy to see that all that is needed is to show that

$$\arg\max_{\pi} \psi(\pi) \cdot -\hat{\psi} = \arg\max_{\pi} \psi(\pi) \cdot \bar{\mathbf{w}}^{\text{SMP}}_{\Pi^n}.$$

To show this, first observe that $\bar{\mathbf{w}}^{\text{SMP}}_{\Pi^n}$ can be also written as

$$\bar{\mathbf{w}}^{\text{SMP}}_{\Pi^n} = \arg\min_{\mathbf{w}\in\mathbb{B}_2} \max_{x\in\Psi^i} w \cdot \psi = \arg\min_{\mathbf{w}\in\mathbb{B}_2} \max_{\psi\in\text{Co}(\Psi^i)} w \cdot \psi, \tag{14}$$

that is, maximizing ψ over $\text{Co}(\Psi^i)$ instead of Ψ^i (SMP). This is correct because for any reward w there is always a maximizer in the convex hull that is one of the vertices (a property of the linear inner product). And therefore, the same maximum value is attained when maximizing over these two sets.

Next, we have that

$$\arg\min_{\psi\in\text{Co}(\Psi^i)} ||\psi||_2^2 = \arg\max_{\psi\in\text{Co}(\Psi^i)} -||\psi||_2^2 \tag{15}$$

$$= \arg\max_{\psi\in\text{Co}(\Psi^i)} -||\psi||_2 = \arg\max_{\psi\in\text{Co}(\Psi^i)} \min_{w\in\mathcal{B}_2} \psi \cdot w. \tag{16}$$

Now, if we denote the optimal solutions to Eq. (16) as $\hat{w}, \hat{\psi}$ then, they are also an optimal solution to Eq. (14) via Von Neuman's min-max theorem. This means that $\bar{\mathbf{w}}^{\text{SMP}}_{\Pi^n} = \hat{w} = -\hat{\psi}/||\hat{\psi}||$.

Thus

$$\arg\max_{\pi} \psi(\pi) \cdot \bar{\mathbf{w}}^{\text{SMP}}_{\Pi^n} = \arg\max_{\pi} \psi(\pi) \cdot -\hat{\psi}/||\hat{\psi}|| = \arg\max_{\pi} \psi(\pi) \cdot -\hat{\psi},$$

where the second inequality follows from the fact that dividing the reward by the same constant across all states does not change the optimal policy (the $\arg\max$).

Finally, note that the function $h = 0.5||x||_2^2$ has 1-Lipschitz gradient and is strongly convex. Thus, since the algorithms are equivalent, Algorithm 2 achieves a linear convergence according to the following theorem.

Theorem 1 (Linear Convergence (Jaggi and Lacoste-Julien, 2015)). *Suppose that h has L-Lipschitz gradient and is μ-strongly convex. Let $D = \{d_\pi, \forall \pi \in \Pi\}$ be the set of all the state occupancy's of deterministic policies in the MDP and let $\mathcal{K} = Co(D)$ be its Convex Hull. Such that $\mathcal{K}$ a polytope with vertices D, and let $M = diam(\mathcal{K})$. Also, denote the Pyramidal Width of D, $\delta = PWidth(D)$ as in (Jaggi and Lacoste-Julien, 2015, Equation 9 1).*

Then the suboptimality h_t of the iterates of all the fully corrective FW algorithm decreases geometrically at each step, that is

$$h(x_{t+1}) \le (1-\rho)h(x_t)\text{ , where } \rho = \frac{\mu\delta^2}{4LM^2}$$

C. Additional implementation details and hyper parameters

When we add a new policy, π^t, to the set Π^{t-1}, we reset the maximum value $v_e^* = \max\{v_e^*, v^t\}$. This step is useful because the policies and their value functions are computed approximately in practice and in some of the domains the optimal performance is not achieved in the first iteration of Algorithm 1.

To bound the intrinsic rewards we first use the following transformation $\tilde{r}_w(s) = \frac{w\cdot\phi(s)\ +\|w\|^2}{\|w\|^2}$ and then apply the following non-linear transformation:

$$r(s) = \left(1 - \exp\left(-\tau \tilde{r}_w(s)\right)\right) / (1 - \exp(\tau)), \tag{17}$$

This transformation is useful when we want the reward to be more sensitive to small variations of the inner product, i.e., when many policies are relatively similar to each other.

Finally, Table 1 summarizes the hyperparameters that we use in Algorithm 1

Table 1: Hyperparameters table

Parameter	Value
Optimality level α (Eq. (8))	0.9
Environment steps per policy	10^6
Number of policies	8
Lagrange entropy regularization weight a_h (Eq. (9))	0.01
Lagrange learning rate	0.1
Lagrange update frequency (N_λ)	30
Estimation decay factor a_d	0.9
Normalization temperature τ (Eq. (17))	3

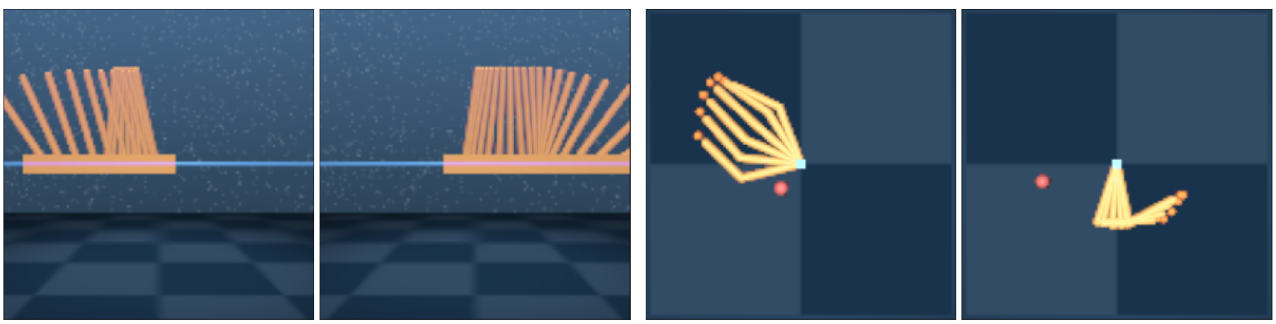
D. Additional results

Our "motion figures" were created in the following manner. Given a trajectory of frames that composes a video $f_1, \ldots, f_T$, we first trim and sub sample the trajectory into a point of interest in time: $f_n, \ldots, f_{n+m}$. We always use the same trimming across the same set of policies (the sub figures in a figure). We then sub sample frames from the trimmed sequence at frequency $1/p$: $f_n, f_{n+p}, f_{n+2p} \ldots,$. After that, we take the maximum over the sequence and present this "max" image. In Python, this simply corresponds to, for example, to

```
n=400, m=30, p=3
indices = range(n,n+m,p)
im = np.max(f[indices])
```

This creates the effect of motion in single figure since the object has higher values then the background.

D.1. Clockwise Diversity in Cartpole and Reacher



(a) Cartpole (b) Reacher

Figure 5: Clockwise Diversity in Cartpole and Reacher.

D.2. Walker Stand

Figure 6: Min

Figure 7: Average

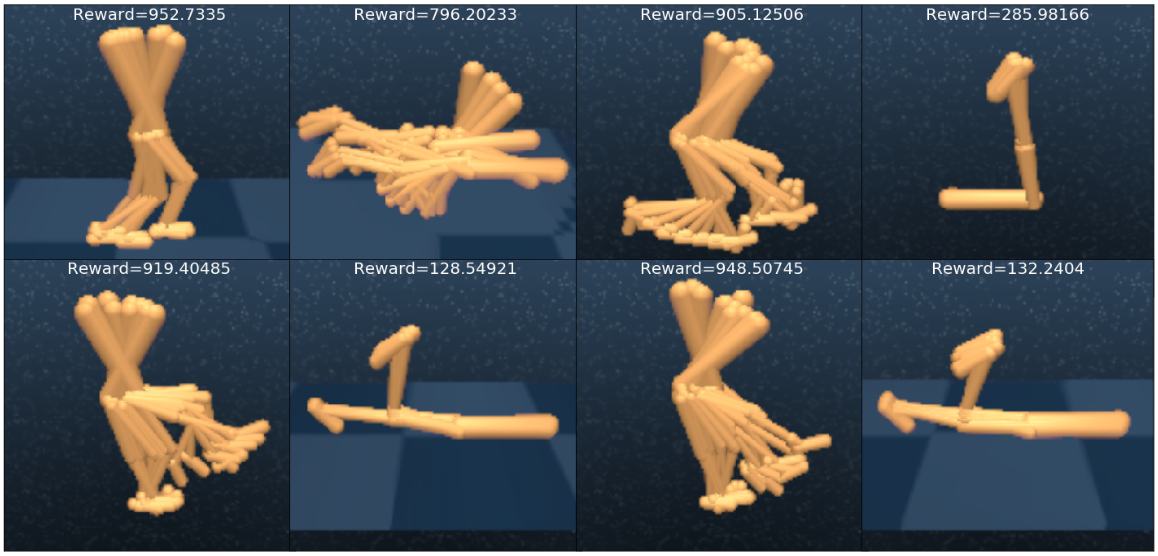


Figure 8: Robustness

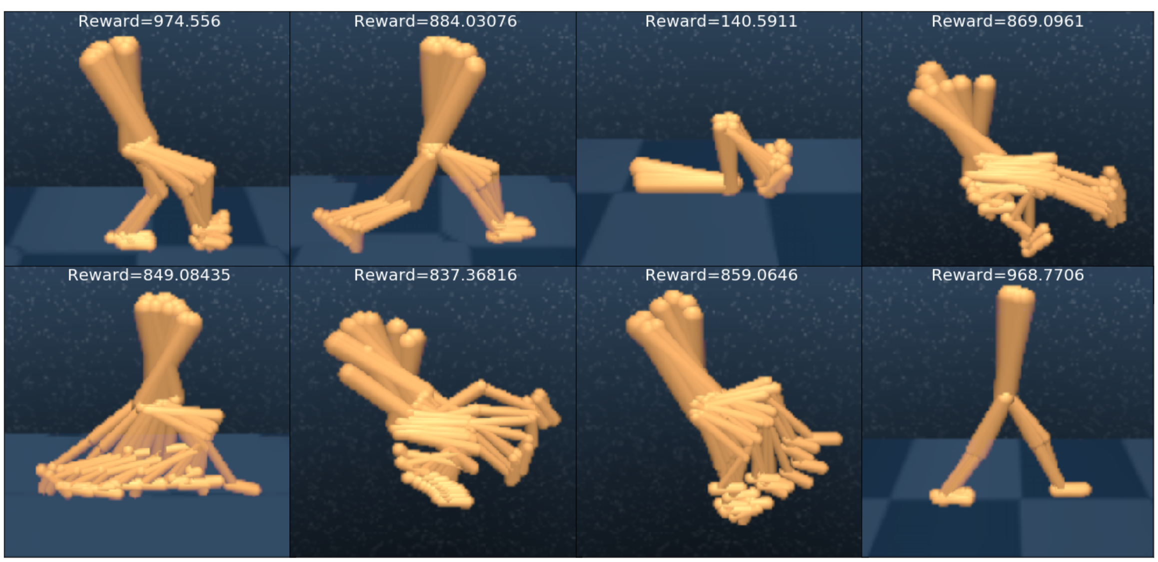


Figure 9: Discrimination

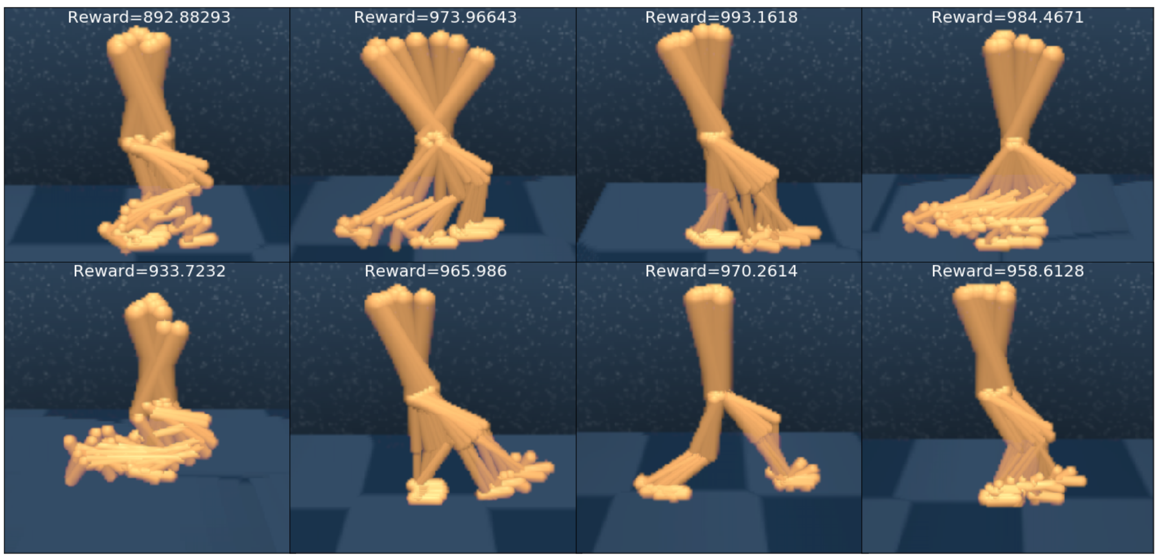


Figure 10: None

D.3. Walker Walk

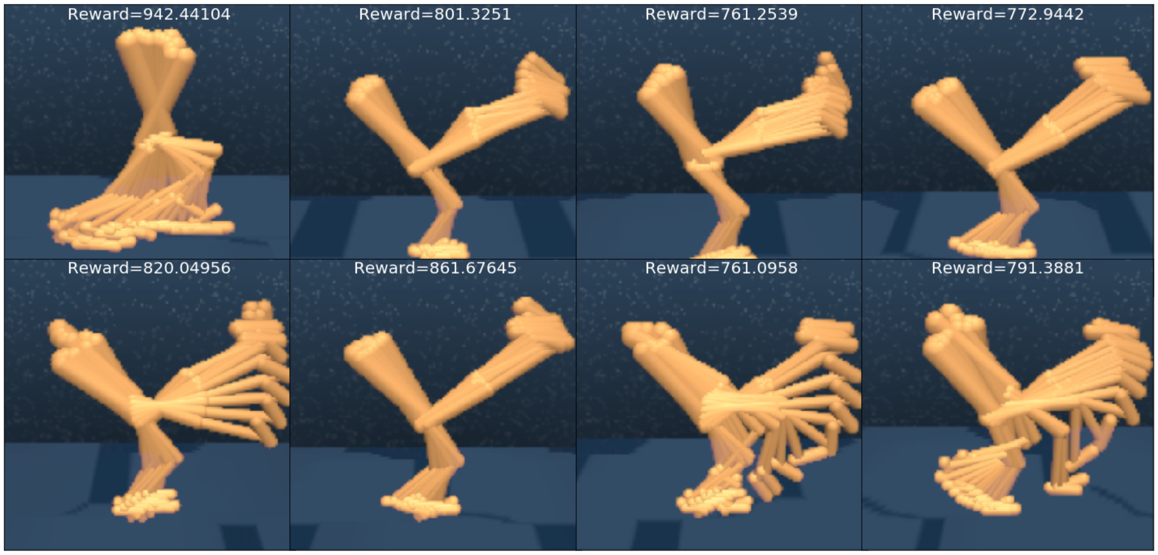


Figure 11: Min

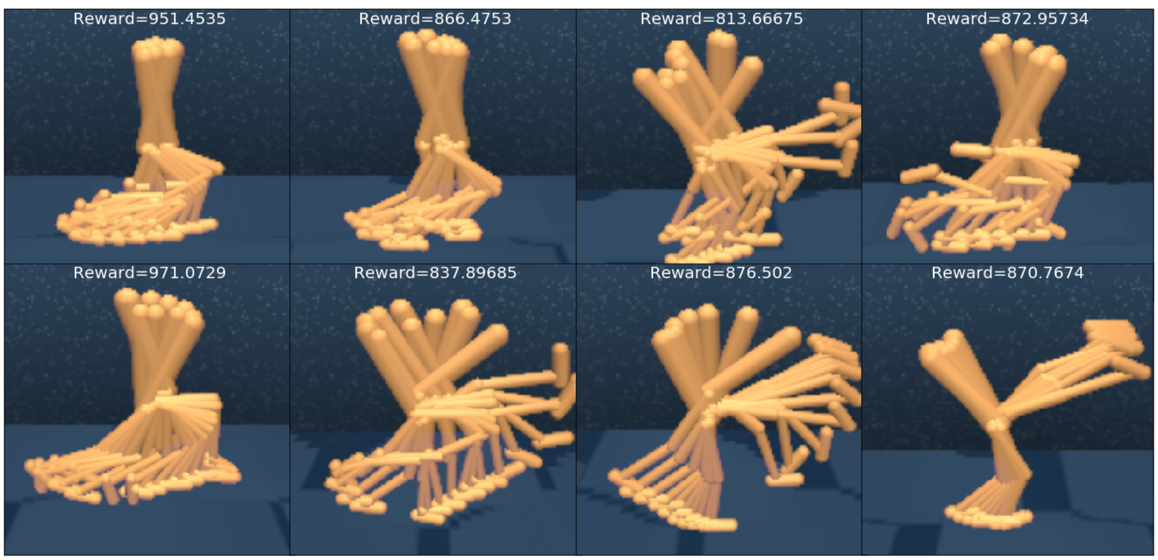


Figure 12: Average

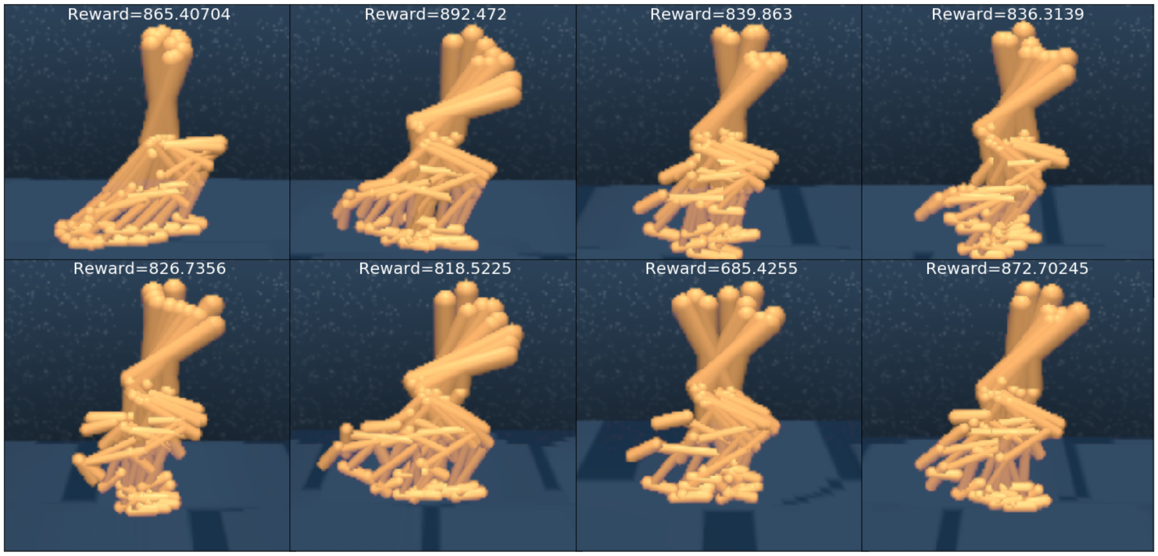


Figure 13: Robustness

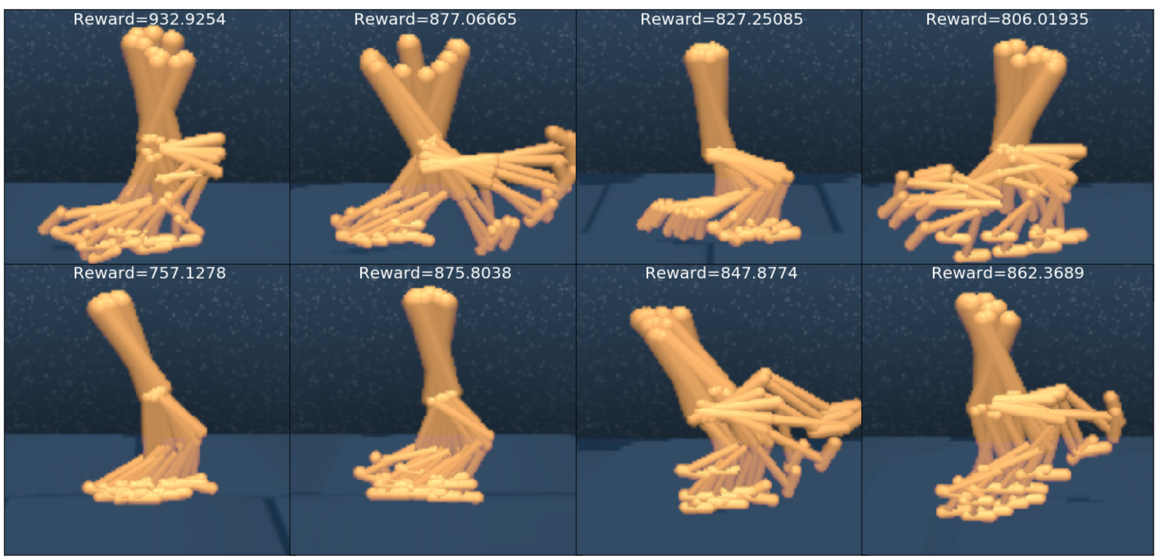


Figure 14: Discrimination

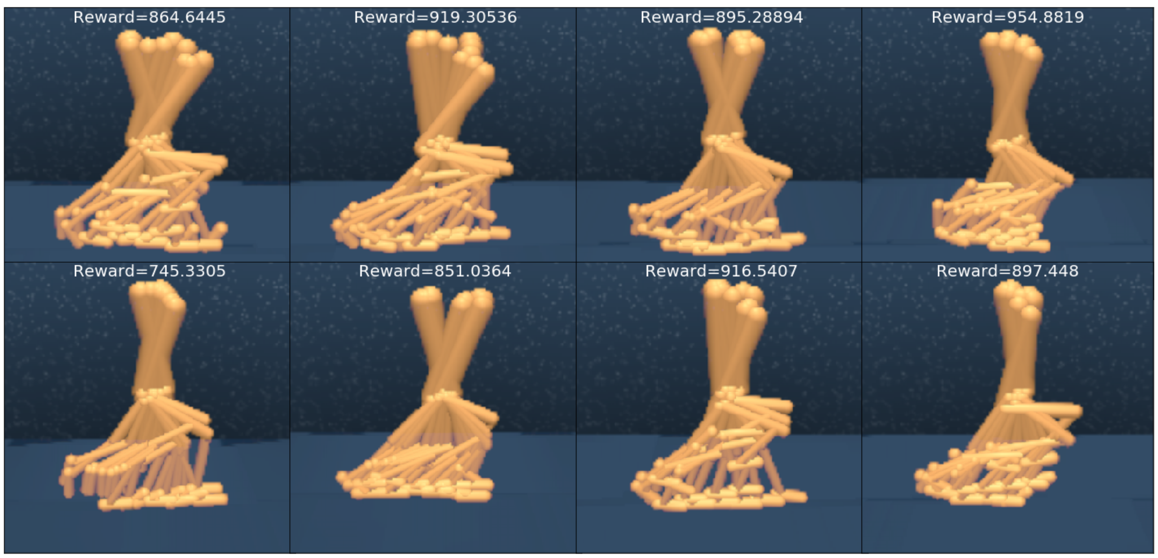


Figure 15: None

D.4. Robustness in Walker

Figure 16: Min

Figure 17: Average

Figure 18: Discrimination

Figure 19: None

D.5. Robustness in Cheetah

Figure 20: Min

Figure 21: Average

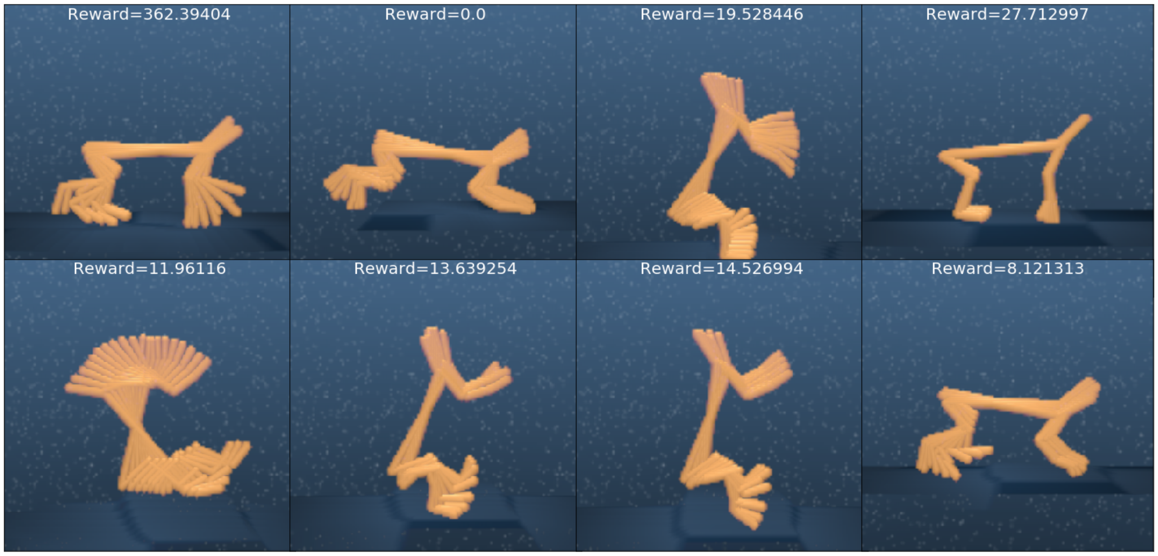


Figure 22: Discrimination

Figure 23: None