Hierarchical Relative Entropy Policy Search

Christian Daniel[1], **Gerhard Neumann**[1], **Jan Peters**[1,2]
[1] Technische Universität Darmstadt, Hochschulstr. 10, 64289 Darmstadt, Germany
[2] Max Planck Institute for Intelligent Systems, Spemannstr. 38, 72076 Tübingen, Germany
{daniel, neumann, peters} @ias.tu-darmstadt.de

Abstract

Many real-world problems are inherently hierarchically structured. The use of this structure in an agent's policy may well be the key to improved scalability and higher performance. However, such hierarchical structures cannot be exploited by current policy search algorithms. We will concentrate on a basic, but highly relevant hierarchy — the 'mixed option' policy. Here, a gating network first decides which of the options to execute and, subsequently, the option-policy determines the action.

In this paper, we reformulate learning a hierarchical policy as a latent variable estimation problem and subsequently extend the Relative Entropy Policy Search (REPS) to the latent variable case. We show that our Hierarchical REPS can learn versatile solutions while also showing an increased performance in terms of learning speed and quality of the found policy in comparison to the non-hierarchical approach.

1 Introduction

In recent years, policy search methods have received a lot of attention due to their strong convergence guarantees [1], their ease of use for function approximation [2], the improved possibilities of incorporating domain knowledge [3] and several impressive application results [4, 5, 6, 7]. A variety of successful policy search methods have been introduced including pair-wise policy comparisons [8], policy gradient methods [9, 2], natural policy gradient methods [1, 10], probabilistic policy search approaches based on EM [7], or based on probabilistic modeling [11]. These policy search methods have been particularly successful in the domain of robot movement generation [12, 13, 14].

However, current methods cannot exploit the hierarchical structure inherent to many real-world problems. Introducing such structures has the potential to improve scalability as well as performance of policy search algorithms [15]. For example, many problems require learning a 'mixed option' policy. That is, given a set of parametrized options, also called motion templates [16], a gating network first determines the option to execute and, subsequently, a continuous action-selection policy determines the parametrization of the option. Thus, in contrast to concentrating on a single parameterized policy (or option), our approach concentrates on representing a mixture of different options, and hence, we are able to represent versatile solutions and not just a single mode of the solution space.

In this paper, we extend the Relative Entropy Policy Search (REPS) algorithm [17] to the hierarchal case, which we denote as HiREPS. REPS regularizes the policy search problem in an information theoretic way. Such regularization was suggested recently from several different perspectives. Bagnell & Schneider [1] showed that the natural policy gradient corresponds to an update where the loss of information is regularized to a fixed number. Peters et. al. [17] subsequently introduced the REPS algorithm which represents a closed form solution to the resulting policy search problem. Independently, Still & Precup [18] showed that the maximum information gain policy update results in the same solution and Azar et al. [19] showed that this update can also be understood as punishing the distance between the controlled system and the uncontrolled one. Finally, Rawlik et al. [20] showed that even previous probabilistic policy search approaches are closely related. All these different arguments have lead to similar solutions despite their different motivations, and the resulting algorithms work well on benchmark problems.

Appearing in Proceedings of the 15^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2012, La Palma, Canary Islands. Volume XX of JMLR: W&CP XX.

We will treat the problem of learning a hierarchical policy as a latent variable estimation problem. For the policy update, we assume that we can only observe the resulting actions of the old policy. The underlying hierarchy is unknown and, therefore, unobserved. Hence, we first estimate the probability that an action has been created by the individual options and, subsequently, weight the update of the options by these probabilities. The resulting algorithm is closely related to expectation maximization (EM) for latent variable models. We prove that such EM mechanisms can be incorporated in the information theoretic regularization of the REPS algorithm in order to get a lower bound of the original optimization problem, which can subsequently be optimized in closed -form.

Furthermore, we introduce an additional constraint into the optimization problem that bounds the uncertainty of identifying an option given an action. As a consequence, the options are separated in the action space, which allows for finding more versatile solutions and also alleviates the problem of averaging over several modes in the solution space, which is present in many current policy search algorithms [21].

Such versatile solutions are more robust to changes in the environment or the robot. For example, if one solution turns out to be unavailable in a new scenario (e.g., due to a damaged part of the robot or a non-stationary environment), we can still opt for the second solution.

2 Hierarchical Relative Entropy Policy Search

We start with the definition of a 'mixed' option policy. Subsequently, we briefly review the Relative Entropy Policy Search algorithm as well as extend it to the latent variable case. We treat the hierarchical structure of the policy as a latent variable, which results in the Hierarchical Relative Entropy Policy Search (HiREPS) algorithm.

2.1 Problem Statement & Notation

We assume the Markov decision process (MDP) framework [22] where an agent is in a state $s \in \mathbb{S}$ and takes an action $a \in \mathbb{A}$. Based on the combination of state and action, the agent transfers to a next state s' and receives a reward $r \in \mathbb{R}$. The goal of the agent is to optimize the expected average rewards. The state transfer happens in accordance to a transition probability distribution $p(s'|s,a) = \mathcal{P}^a_{ss'}$. In real-world robotics problems, both states and actions, are frequently continuous.

In addition to the MDP assumptions, we require that the behavior is composed of episodes, where each episode consists of a series of actions. Such consistency can be assured by grouping sequences of actions into options [23]. As a result, our policy π, which maps states to actions, has to be based on these options $o \in \mathbb{O}$. The behavior during option o is governed by the (sub-)policy $\pi(a|s,o)$. This sub-policy is activated by a supervisory policy $\pi(o|s,o')$ where o' denotes the option that was active in the previous time step in order to account for the temporal consistency. Following the classical options framework as described in [23], $\pi(o|s,o')$ can be decomposed into the termination probability β of option o' and the gating network $\pi(o|s)$, i.e.,

$$\pi(o|s,o') \propto \beta(o',s)\pi(o|s) + \mathbb{1}_o(o')(1-\beta(o',s)),$$

where $\mathbb{1}_o$ is the indicator function that is one when $o' = o$ and zero otherwise.

However, as the estimation of the temporal consistency (i.e., $\beta(o',s)$) is a difficult problem by itself, we will limit ourselves to the case of options which last exactly one time step, i.e., $\pi(o|s,o') = \pi(o|s)$. Note that this case also includes the motion template framework [16] which works on the level of abstract steps. Here, a step takes as long as a predefined parametrized motion template is executed.

The resulting policy $\pi(a|s)$ in this setup can be expressed as a 'mixture of options' policy, i.e.,

$$\pi(a|s) = \sum_o \pi(o|s)\pi(a|o,s).$$

The expected reward is hence given by

$$J(\pi) = \sum_{s,a,o} \mu^\pi(s)\pi(o|s)\pi(a|s,o)r(s,a) \qquad (1)$$

where $\mu^\pi(s)$ denotes the state distribution. We assume that $\mu^\pi(s)$ is a stationary state distribution defined by

$$\forall s' : \sum_{s,a} \mu^\pi(s)\pi(a|s)p(s'|s,a) = \mu^\pi(s'), \qquad (2)$$

which holds under mild conditions [2]. Here, we will only require that this condition holds for state features $\boldsymbol{\phi}(s)$, i.e.,

$$\sum_{s'} \boldsymbol{\phi}(s') \sum_{s,a} \mu^\pi(s)\pi(a|s)p(s'|s,a) \qquad (3)$$
$$= \sum_{s'} \mu^\pi(s')\boldsymbol{\phi}(s').$$

The goal of this paper is to develop an algorithm that finds both, a good supervisory policy $\pi(o|s)$ and good sub-policies $\pi(a|s,o)$.

2.2 Relative Entropy Policy Search

Several authors [19, 17, 18] have argued from different perspectives (i.e., steps away from a natural dynamics distribution [19], loss of information [17] and optimal exploration [18]) that after a policy update, the new state-action distribution $p(a, s)$ should remain close to a reference distribution $q(a, s)$, which could be the uncontrolled dynamics [19], the previous policy [17], the uniform distribution [18] or maybe a distribution obtained by watching an expert performing a task.

Closeness among probability distributions is frequently measured using the Kullback-Leibler divergence $D_{\mathrm{KL}}\left(p(a,s)||q(a,s)\right) = p(a,s)\log\left(p(a,s)/q(a,s)\right)$.

In REPS, the distribution $p(a, s) = \mu(s)\pi(a|s)$ is now defined as the distribution with maximum reward while bounding $D_{\mathrm{KL}}\left(p(a,s)||q(a,s)\right) \leq \epsilon$ and ensuring the consistency of the steady state distribution, as shown in Equation (3). The distribution $p(s, a)$ can be found in closed form and is given by

$$\begin{aligned} p(a,s) &\propto q(a,s)\exp\left(\frac{\delta_{\mathrm{sa}}}{\eta}\right), \qquad (4)\\ \delta_{\mathrm{sa}} &= R_{\mathrm{sa}} + \sum_{s'} \mathcal{P}^a_{ss'} V(s') - V(s), \end{aligned}$$

where δ_{sa} denotes the Bellman error and $V(s) = \boldsymbol{\theta}^T\boldsymbol{\phi}(s)$. The parameters η and $\boldsymbol{\theta}$ denote Lagrange multipliers and can efficiently be found by minimizing the convex dual-function $g(\eta, \boldsymbol{\theta})$ of the original optimization problem.

REPS with Parametric Distributions

In the continuous case, the distributions $\mu(s)$ and $\pi(a|s)$ need to be represented by parametric distributions such as Gaussians or linear Gaussian models. This representation can be obtained by minimizing the KL-divergence $D_{\mathrm{KL}}\left(p(a,s)||\mu(s)\pi(a|s)\right)$ which is given by

$$\begin{aligned} &= \int_{s,a} p(a,s)\log\big(\mu(s)\pi(a|s)\big)dsda + \mathrm{const} \\ &\approx \sum_{(s,a)\sim q(s,a)} \exp\left(\frac{\delta_{\mathrm{sa}}}{\eta}\right)\log\big(\mu(s)\pi(a|s)\big). \qquad (5) \end{aligned}$$

The distribution $q(a, s)$ does not need to be known. For both minimizing D_{KL} as well as optimizing the dual function g, we only require access to samples from this distribution. Also, note that the KL-divergence minimization as defined in Equation (5) is a simple operation — it is equivalent to calculating the weighted maximum likelihood estimates of the parameters of π.

2.3 REPS with Latent Variables

In the hierarchical case, we also have to incorporate our options o. However, the optimal hierarchy of q is typically unknown and, therefore, o cannot be observed , i.e., we only have access to samples from the marginal $q(s, a)$. Thus, we still bound the marginals $p(a, s) = \sum_o p(s, a, o)$ and treat o as latent variable. Doing so, however, is not trivial as the log of the marginal does not allow for a closed form solution. Instead, we can reformulate the problem using the relation $p(s, a) = p(s, a, o)/p(o|s, a)$ which leads us to the following bound

$$\sum_{s,a}\sum_{o} p(s,a,o)\log\frac{p(s,a,o)}{q(s,a)p(o|s,a)} \leq \epsilon. \qquad (6)$$

Having this bound does not help us directly as the conditional $p(o|s, a)$ also depends on the marginal. However, we can use this bound in an iterative expectation-maximization (EM)-like manner.

- In the *expectation (E) step*, we estimate a proposal distribution $\tilde{p}(o|s, a)$ for $p(o|s, a)$. In order to do so, we fix the current model distribution $p(s, a, o)$ and calculate $\tilde{p}(o|s, a) = p(s, a, o)/p(s, a)$. Note that the distribution $\tilde{p}(o|s, a)$ is often referred to as the responsibility in EM-based algorithms.
- In the *maximization (M) step*, we use $\tilde{p}(o|s, a)$ instead of the real model $p(o|s, a)$. Using a fixed distribution $\tilde{p}(o|s, a)$ for the bound defined in Equation (6) allows for a closed form solution.

In the appendix, we show that both the E- and the M-step indeed maximize a lower bound of the original optimization problem, which is tight after each E-step.

Learning Versatile Solutions

In the hierarchical setting of REPS, we are often interested in a versatile solution space. Therefore, it makes sense to learn options which are clearly separable in the action space. In order to do so, we also bound the expected entropy of the conditionals $p(o|s, a)$.

$$-\sum_{s,a} p(s,a)\sum_{o} p(o|s,a)\log p(o|s,a) \leq \kappa, \qquad (7)$$

This bounding ensures that the options are clearly identifiable (given a state and an action) and do not overlap. Again, we will replace the term $\log p(o|s, a)$ by the responsibilities $\log \tilde{p}(o|s, a)$ and add it to the constraints of our optimization problem. The optimization problem using $\tilde{p}(o|s, a)$ still defines a lower bound of the original problem (see the appendix). We will always choose κ as percentage of the entropy of the current responsibilities $\tilde{p}(o|s, a)$ such that we can gradually decrease the entropy of the options.

2.4 Resulting Policy Updates

The combination of the objectives in Equations (1, 2, 6 and 7) as stated above, yields a well-formulated optimization problem given below.

Problem Statement. *Find a gating policy $\pi(o|s)$ and an option policy $\pi(a|s,o)$ such that the resulting policy $\pi(a|s) = \sum_o \pi(o|s)\pi(a|s,o)$ maximizes the expected return $J(\pi)$ while bounding the information loss by ϵ, i.e.,*

$$\begin{aligned}
\max_{\pi,\mu^\pi} J(\pi) &= \sum_{s,a,o} \mu^\pi(s)\pi(a,o|s)R_{sa}, \\
\text{s.t.} \quad \epsilon &\geq \sum_{s,a,o} p(s,a,o) \log \frac{p(s,a,o)}{q(a,s)\tilde{p}(o|s,a)}, \\
\kappa &\geq -\sum_{s,a} p(s,a) \sum_o p(o|s,a) \log \tilde{p}(o|s,a), \\
\sum_{s'} \boldsymbol{\phi}(s')\mu^\pi(s') &= \sum_{s'} \sum_{s,a,o} \mathcal{P}^a_{ss'} \mu^\pi(s)\pi(a|o,s)\pi(o|s)\boldsymbol{\phi}(s'), \\
1 &= \sum_{s,a,o} \mu^\pi(s)\pi(a,o|s). \qquad (8)
\end{aligned}$$

Here, $\pi(a,o|s) = \pi(o|s)\pi(a|s,o)$ denotes the joint policy over actions and options and $p(s,a,o)$ is given by $\mu(s)\pi(o|s)\pi(a|s,o)$.

The resulting joint distribution is then given by

$$p(s,a,o) \propto q(s,a)\tilde{p}(o|s,a)^{1+\xi/\eta} \exp\left(\frac{\delta_{\text{sa}}}{\eta}\right). \qquad (9)$$

The parameters η, ξ, and $\boldsymbol{\theta}$ (contained in δ_{sa}) are again calculated by optimizing the convex dual function $g(\boldsymbol{\theta}, \eta, \xi)$ which is also given in the appendix. The state distribution, the gating network, and the sub-policies can be determined from $p(s,a,o)$ by minimizing $D_{\text{KL}}\left(p(a,s,o)||\mu(s)\pi(o|s)\pi(a|s,o)\right)$. By setting the number of options to 1, the resulting equations correspond to the standard derivation of REPS. In this case, κ does not have any influence on the optimization process.

2.5 Episodic HiREPS

In this paper, we will concentrate on the episodic case of HiREPS, where only one parametrized option is executed until the episode is terminated. In the episodic case, we do not have a steady-state distribution — the state distribution $\mu_0(s)$ now denotes the distribution of the initial states. Instead of the steady-state distribution constraint, we now require that the expected state features $\sum_{s,a,o} p(s,a,o)\boldsymbol{\phi}(s)$ match the observed state features $\hat{\boldsymbol{\phi}}$ of the initial state distribution. The resulting policy of this optimization problem is now given by

$$p(s,a,o) \propto q(s,a)\tilde{p}(o|s,a)^{1+\xi/\eta} \exp\left(\frac{R_{\text{sa}} - \boldsymbol{\theta}^T\boldsymbol{\phi}(s)}{\eta}\right),$$

where $\boldsymbol{\theta}^T\boldsymbol{\phi}(s)$ accounts for the value of the initial states s.

We are now ready to provide the episodic HiREPS algorithm which can be seen in Table 1.

Input: Information loss tolerance ϵ, Entropy tolerance κ, Number of options n
Initialize π using n Gaussians with random mean
while not converged
Set sample policy: $q(a\|s) = \sum_o \pi_{\text{old}}(o\|s)\pi_{\text{old}}(a\|s,o)$
Sample: collect samples from the sample policy $\{s_i \sim p(s_0), a_i \sim q(a\|s_i), R_i\}_{i\in\{1,..,N\}}$
Proposal distribution: $\tilde{p}(o\|s_i,a_i) = p_{\text{old}}(o\|s_i,a_i)$
Minimize the dual function $[\boldsymbol{\theta},\eta,\xi] = \arg\min_{[\boldsymbol{\theta},\eta,\xi]} g\left(\boldsymbol{\theta},\eta,\xi\right)$
Policy update: *Calculate weighting* $p(s_i,a_i,o) \propto \tilde{p}(o\|s_i,a_i)^{1+\xi/\eta} \exp\left(\frac{\mathcal{R}_i - \boldsymbol{\theta}^T\boldsymbol{\phi}(s)}{\eta}\right)$ *Estimate distributions* $\pi(o\|s)$ and $\pi(a\|s,o)$ by weighted ML estimates
Output: Policy $\pi(a,o\|s)$

Table 1: Episodic HiREPS

Illustration

We illustrate our algorithm on a simple toy task, where we have a two dimensional action space and a bi-modal reward function (see Figure 1). We will refer to this task as the Two-Gaussians Task for the remainder of this paper. In this task, the reward distribution consists of two Gaussians such that the we have two global optima.

Examining the Two-Gaussians Task is interesting for two reasons: first, many standard policy search methods have problems with multi-modal solution spaces, i.e., they will be drawn to all optima in multi-modal tasks and, therefore, converge slowly [21]. Here we will see that HiREPS solves this problem by bounding the entropy of the options. Second, as the task has multiple solutions, we would like to represent a versatile solution space by learning all modes of the reward function.

In Figure 1, we show the results of comparing our algorithm qualitatively to the standard REPS algorithm and to HiREPS without bounding of the options' en-

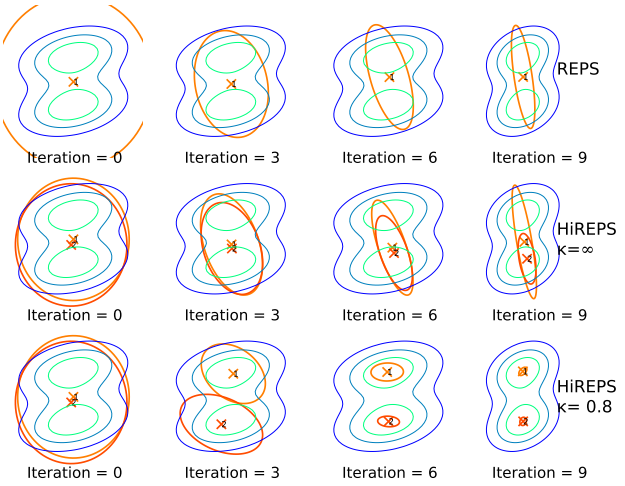


Figure 1: Schematic sketch of the behavior of the REPS, the HiREPS with and without bounding of the options' entropy ($\kappa = 0.8, \kappa = \infty$) . Blue and green lines show the contours of the bi-modal reward function. The REPS and the HiREPS without bounding try to average over both modes. After many iterations, they will eventually converge to one (or two) mode(s). The HiREPS with $\kappa = 0.8$ separates both options and is therefore able to find both modes reliably.

tropy (i.e., $\kappa = \infty$). The standard REPS algorithm tries to average over both modes and, hence, takes a long time to converge. For the HiREPS without bounding the entropy, the behavior is quite similar. All options are attracted by both modes. Thus, in most cases, both options find the same mode. In cases where they do separate, the separation takes a long time. When using HiREPS with a bound on the entropy, the options quickly separate and concentrate on different modes, allowing for a fast improvement of the policy without getting stuck between two modes. The illustrations in Figure 1 represent typical cases when applying the previously introduced algorithms to this setup.

3 Evaluation

In order to evaluate our algorithm, we choose two tasks, i.e., a puddle world task and a robot Tetherball task. For all experiments, we use Gaussian gating networks and linear Gaussian models for the sub-policies. The gating networks and the sub-policies are randomly initialized. Whenever the prior $\pi(o|s)$ of an option gets too small (i.e., $\pi(o) < 10^{-4}$) we delete the option. However, in order to avoid options getting deleted too quickly, we assure that each option gets a minimum amount of samples in the sampling process. We also bound the minimum variance of our Gaussian models to small values in order to avoid singularities. All experiments were averaged over 20 trials. After the sampling process, we always evaluate the quality of the exploration-free policy (i.e., without variance) found so far. We also evaluate the versatility of the found solutions. For all setups with states, we will use a squared exponential kernel as basis functions for the state features, i.e

$$\phi_i(\mathbf{s}) = \exp\left(-\frac{1}{2}(\mathbf{s}-\mathbf{s}_i)^T\mathbf{\Lambda}^{-1}(\mathbf{s}-\mathbf{s}_i)\right),$$

where $\mathbf{\Lambda}$ is a diagonal matrix denoting the bandwidth of the kernel. We will compare our algorithm to the non-hierarchical counterpart, REPS [17], which is equivalent to our algorithm with just one option. Both algorithms receive the same number of samples per iteration. We will also investigate the effect of bounding the entropy of the options on the performance of the algorithm. For all presented results, we have optimized the parameters of the algorithms to deliver the best performance.

3.1 PuddleWorld Experiment

In a first toy task, we test HiREPS on a variation of the puddle world as seen in [24]. Our version differs, as the action space is continuous instead of discrete. We use Dynamic Movement Primitives [25] to represent the options. An option is given by two DMPs, one for each of the x and y dimensions. We assume that the goal is known and, therefore, fix the point-attractor of the DMPs at the known goal position. For the x-dimension, the DMP is fixed, but for the y-dimension we use five basis-functions of the DMPs to modify the trajectory. The linear weights of these basis-functions define the parameters of the option. Thus, the action vector $\mathbf{a}$ of our options is five-dimensional.

The reward of the task is given by the negative length of the line segments, which encourages shorter solutions. An additional punishment occurs for passing through the puddles. The arrangement of the puddles can be seen in Figure 2. The presented puddle world has two solutions which are located close to each other, however, the mean of both solutions leads through a puddle and, therefore, yields lower rewards.

In Figure 3, we evaluate the performance of REPS and HiREPS with and without bounding of the options' entropy. For HiREPS, just two options were used. REPS takes a long time to reliably find good solutions, as the algorithm averages over both modes. HiREPS without bounded entropy performs slightly better than REPS. However, the advantage of HiREPS is much more pronounced when bounding the entropy. Furthermore, if

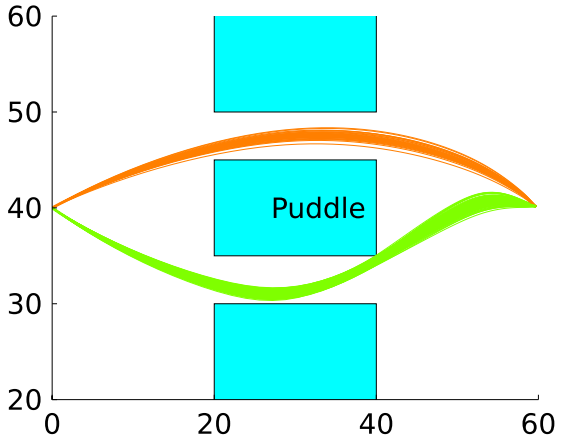


Figure 2: The puddle world task. Sub-policies are represented as two-dimensional DMPs with fixed end points. The DMPs have five basis functions per dimension and we learn the weights of the basis functions for the y dimension while leaving the weights for the x dimension fixed. The plot shows trajectories sampled from two options found after 80 iterations of HiREPS.

we do limit the entropy, the algorithm is able to reliably find both modes.

3.2 Simulated Tetherball Experiment

Our aim is to adapt the human game Tetherball for a robotic player. Specifically, the task consists of a ball, an elastic rope, an obstacle (i.e., a pole) and a target. The ball is hung in front of the pole using the elastic rope. The pole is placed on a line between the ball's resting position and the goal, such that it is impossible to hit the target with a single punch of the ball in direction of the target. The goal of the robot is to hit the ball such that it comes as close

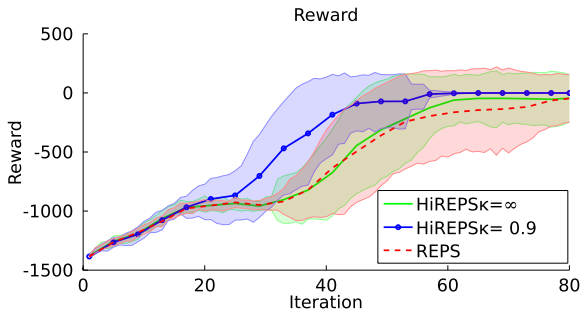
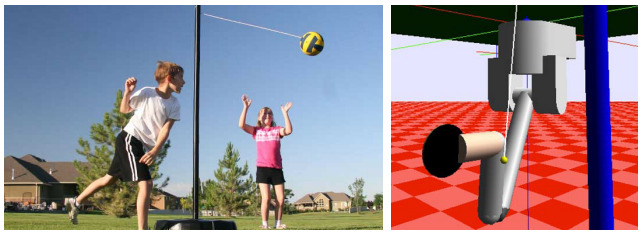


Figure 3: Performance of HiREPS and REPS on the puddle world task. As HiREPS can represent both solutions it does not get stuck averaging over both modes. Note, that this effect is much more pronounced if we bound the entropy of the options.



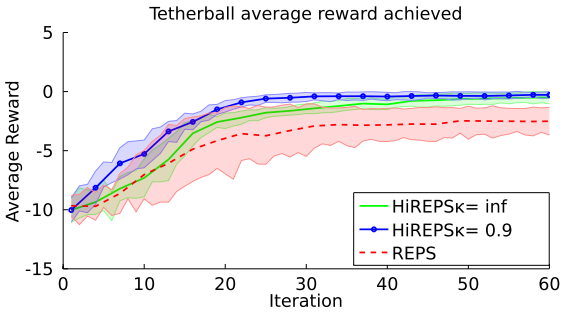
(a) Tetherball (b) Simulation of robot Tetherball

Figure 4: (a) A standard game of Tetherball. Two opponents play against each other. The goal is to hit the ball to the partner. Adapted from [26] with permission. (b) The simulation of the robot Tetherball task. The goal is to hit the ball to a fixed target.

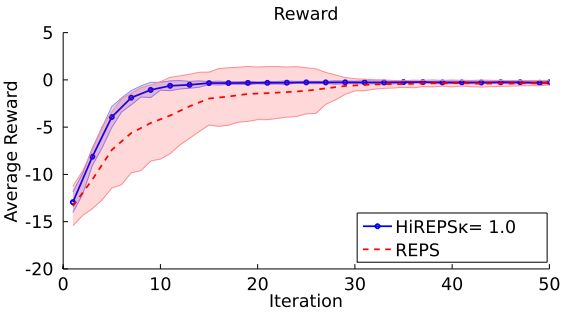
to the target as possible. In order to hit the target, the robot has to exploit the elasticity of the rope and bouncing of the ball against the obstacle. On its' path, the ball may bounce off of the pole and the rope may wind around the pole. Note that this task presents a versatile solution space, as many different strategies successfully hit the target. Our goal is to model this versatile solution space with HiREPS.

For the purpose of testing the HiREPS method, we use a strongly simplified setup where the Tetherball task is implemented in a Matlab simulation. We initialize HiREPS with 30 randomly located options. The agent can push the ball with a two dimensional impulse $\{F_x, F_y\}$. The reward of a push is given by the negative minimum squared distance of the ball to the target throughout the ball's trajectory. The initial state of the agent is given by the initial position of the ball before hitting it. We only vary the x-position of the ball and learn different solutions to hit the target. We again compare HiREPS with and without bounding to the standard REPS approach. In Figure 5a, we evaluate the average reward of all three approaches. The results show that HiREPS with a bound on the entropy outperforms the two other methods. The HiREPS approach without bounding of the entropy is clearly better than the REPS approach, as REPS only uses one option to cover the whole state space. Thus, REPS needs to approximate the optimal policy using a single linear model. Therefore, in order to have a fair comparison to REPS, we also compare HiREPS and REPS on the Tetherball task without states i.e., we always start from the same initial state in the middle. This comparison can be found in Figure 5b. We can observe that REPS is impaired by the multi-modality of the solution space as it tries to average over several modes.

In Figure 6a, we show the number of options used for

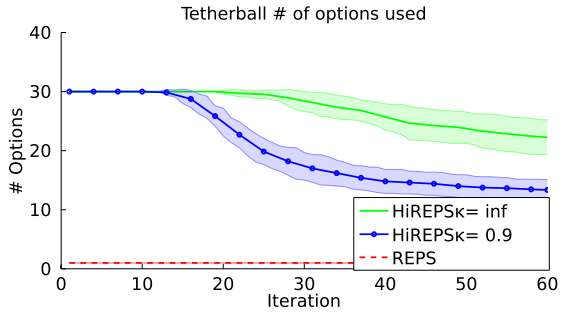


(a) Tetherball Average Reward

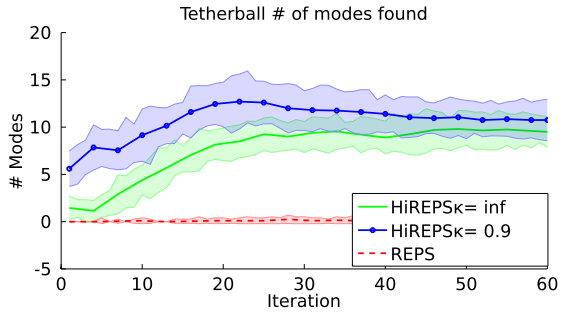


(b) Tetherball without States, Average Reward

Figure 5: (a) Average reward gathered by the REPS and the HiREPS with and without bounding the entropy of the options in the Tetherball task including states. As REPS only uses a single option and thus a linear model as policy, it cannot represent the complicated structure of the solution. The HiREPS approach benefits from bounding the entropy of the options. (b) Average reward in the Tetherball task without states. While REPS also finds the optimal solution, HiREPS benefits from its structured policy representation and outperformes REPS in learning speed.



(a) Tetherball Number of Options Used



(b) Tetherball Number of Modes Found

Figure 6: (a) Number of options used by the HiREPS approach with and without bounding the entropy. If the prior $p(o)$ of an option becomes too small, the option gets deleted. With the bounding of the entropy, less options are used while the performance of the algorithm is increased. (b) Number of modes found by both approaches. We can see that even though HiREPS with bounding the entropy uses less options, it can find more modes than HiREPS without bounding the entropy of the options.

different bounds of the entropy κ. By bounding the entropy, we avoid that the options overlap and, hence, superfluous options get deleted more quickly.

As shown in Figure 6b, we also evaluate the number of modes found by the HiREPS with bounding and without bounding the entropy. In order to do so, we divide each dimension of the state action space into 5 partitions and count the number of partitions which contain at least one option with an average reward larger than -1. The plot shows that, because the options distribute more uniformly in the state-action space due to the bounding, we can find more modes. Thus, the bounding also helps us to find more versatile solutions as we avoid situations where multiple options concentrate on the same solution. In Figure 7, we illustrate eight different options found by the HiREPS algorithm. These options represent very different solutions and multiple options can be chosen for a certain state.

4 Conclusion & Discussion

In this paper, we presented a new hierarchical policy search method and integrated it into the relative entropy reinforcement learning (REPS) framework. In order to do so, we extended the REPS framework to the latent variable case, and formulated the problem of estimating the hierarchy of the control policy as a latent variable estimation problem. We could show that, even for basic hierarchical structures such as the 'mixture of options' policy, our algorithm can benefit from exploiting this hierarchy and outperforms the standard REPS framework in terms of both learning speed as well as solution quality.

As we have seen, the 'mixture of options' hierarchy can alleviate a basic deficiency of many policy search algorithms, i.e., averaging over several modes in the

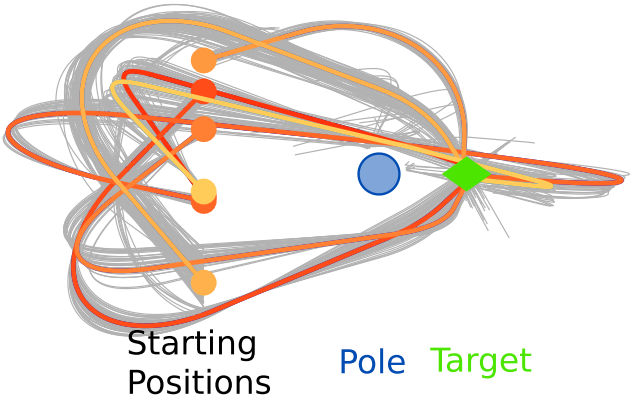


Figure 7: Trajectories for 8 out of 17 options found by HiREPS for the simulated Tetherball task. The agent can choose with which force to shoot the ball, and the objective is to hit the target with the ball. Thick colored lines show the mean of the options, gray lines show samples from different states created by that option.

solution space. In addition, the algorithm offers the appealing perspective of learning versatile solutions by not just concentrating on a single solution, but representing multiple solutions at once.

Currently, our algorithm uses simple hierarchical structures that are insufficient for many continuous tasks. However, we are planning to use more complicated hierarchical structures which capture the temporal consistency of the options or which incorporate continuous latent variables. Furthermore, we are also planning to incorporate importance sampling for efficient sample reuse in order to further increase the sample efficiency of our algorithm. In the future, we will use this method to play real-world Tetherball on a physical Barrett WAM 7-DoF robot arm.

5 Acknowledgements

The authors want to thank for the support of the European Union projects # FP7-ICT-270327 (Complacs) and # 248 273 (GeRT).

A Derivation of the Lower Bound

Consider the optimization problem in Equation (8) with the real conditional $p(o|s,a)$ instead of the responsibilities $\tilde{p}(o|s,a)$. For simplicity, we neglect the steady-state distribution and the normalization constraint, however, our derivation is not affected by these constraints. The Lagrangian of this problem is then given by

$$L(p,\eta,\xi) = \sum_{s,a,o} p(s,a,o) R_{\mathrm{sa}} + \eta\left(\epsilon - \sum_{s,a,o} p(s,a,o)\log\frac{p(s,a,o)}{q(s,a)p(o|s,a)}\right) + \xi\left(\kappa + \sum_{s,a} p(s,a)\sum_{o} p(o|s,a)\log p(o|s,a)\right) \tag{10}$$

Simplifying the terms, we get

$$L(p,\eta,\xi) = \sum_{s,a,o} p(s,a,o)\Big(R_{\mathrm{sa}} - \eta\log\frac{p(s,a,o)}{q(s,a)p(o|s,a)^{1+\xi/\eta}}\Big) + \eta\epsilon + \xi\kappa \tag{11}$$

However, determining a closed form solution for $p(s,a,o)$ is infeasible as the conditional $p(o|s,a)$ is inside the log. Yet, we can determine a lower bound $F(p,\eta,\xi,\tilde{p})$ by using a proposal distribution $\tilde{p}(o|s,a)$ for $p(o|s,a)$ which we can iteratively maximize in an EM-like manner. We need to verify that F is a lower bound on L and that maximizing F w.r.t $\tilde{p}$ is equivalent to setting $\tilde{p}(o|s,a) = p(o|s,a)$, both of which follows from the relation

$$F = L - (\eta+\xi)\sum_{s,a} p(s,a)\underbrace{\sum_{o} p(o|s,a)\log\frac{p(o|s,a)}{\tilde{p}(o|s,a)}}_{D_{KL}\left(p(o|s,a)||\tilde{p}(o|s,a)\right)\geq 0}.$$

After the E-step, the lower bound is tight, i.e., $\max_{\tilde{p}} F(p,\eta,\xi,\tilde{p}) = L(p,\eta,\xi)$. In the M-step, we fix $\tilde{p}$ and maximize F w.r.t p, η and ξ. This defines our constraint optimization problem.

A.1 The Dual Function

The dual-function for the steady-state constraint is given by

$$g(\boldsymbol{\theta},\eta,\xi) = \epsilon\eta + \kappa\xi + \eta\log\left(\textstyle\sum_{s,a,o} q(s,a)\tilde{p}(o|s,a)^{1+\xi/\eta}\exp\left(\frac{\delta_{\mathrm{sa}}}{\eta}\right)\right).$$

For the episodic case, the dual-function is given by

$$\begin{aligned} g(\boldsymbol{\theta},\eta,\xi) &= \epsilon\eta + \kappa\xi + \boldsymbol{\theta}^T\hat{\boldsymbol{\phi}} + \eta\log\left(\sum_{s,a} q(s,a)Z_{\mathrm{sa}}\right) \\ Z_{\mathrm{sa}} &= \sum_{o}\tilde{p}(o|s,a)^{1+\xi/\eta}\exp\left(\frac{R_{\mathrm{sa}} - \boldsymbol{\theta}^T\boldsymbol{\phi}(s)}{\eta}\right). \end{aligned}$$

Both dual-functions are convex in their parameters.

References

[1] J. Bagnell, "Covariant Policy Search," *International Joint Conference on Artificial Intelligence (ICJAI)*, 2003.

[2] R. Sutton, D. McAllester, and S. Singh, "Policy gradient methods for reinforcement learning with function Approximation," *Advances in Neural Information Processing Systems (NIPS)*, 2000.

[3] P. Stone, "Scaling Reinforcement Learning Toward RoboCup Soccer," *International Conference on Machine Learning (ICML)*, 2001.

[4] J. A. Bagnell and J. G. Schneider, "Autonomous Helicopter Control using Reinforcement Learning Policy Search Methods," in *International Conference on Robotics and Automation (ICRA)*.

[5] M. Rosenstein, "Robot Weightlifting by Direct Policy Search," *International Joint Conference on Artificial Intelligence (ICJAI)*, 2001.

[6] N. Kohl, "Policy Gradient Reinforcement Learning for Fast Quadrupedal Locomotion," *International Conference on Robotics and Automation (ICRA)*, 2004.

[7] J. Kober and J. Peters, "Policy Search for Motor Primitives in Robotics," *Machine Learning*, 2008.

[8] M. Strens, "Policy Search Using Paired Comparisons," *Journal of Machine Learning Research (JMLR)*, 2003.

[9] J. Baxter, "Infinite-horizon Policy-gradient Estimation," *Journal of Artificial Intelligence Research (JAIR)*, 2001.

[10] J. Peters and S. Vijayakumar, "Natural Actor-Critic," *European Conference on Machine Learning (ECML)*, 2005.

[11] M. Deisenroth, "PILCO: A Model-based and Data-efficient Approach to Policy Search," in *International Conference on Machine Learning (ICML)*, 2011.

[12] J. Kober, K. Mülling, and O. Krömer, "Movement Templates for Learning of Hitting and Batting," *International Conference on Robotics and Automation (ICRA)*, 2010.

[13] G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng, "Learning CPG-based Biped Locomotion with a Policy Gradient Method: Application to a Humanoid Robot," *International Journal of Robotics Research*, 2008.

[14] P. Kormushev, S. Calinon, and D. G. Caldwell, "Robot Motor Skill Coordination with EM-based Reinforcement Learning," in *International Conference on Intelligent Robots and Systems (IROS)*, 2010.

[15] A. Barto, "Recent Advances in Hierarchical Reinforcement Learning ," *Discrete Event Dynamic Systems*, 2003.

[16] G. Neumann, W. Maass, and J. Peters, "Learning Complex Motions by Sequencing Simpler Motion Templates," in *International Conference on Machine Learning (ICML)*, 2009.

[17] J. Peters, K. Mülling, and Y. Altün, "Relative Entropy Policy Search," in *National Conference on Artificial Intelligence (AAAI)*, 2010.

[18] S. Still and D. Precup, "An Information-theoretic Approach to Curiosity-driven Reinforcement Learning," *International Conference on Humanoid Robotics*, 2011.

[19] M. G. Azar, V. Gomez, and H. J. Kappen, "Dynamic Policy Programming," *arXiv.org*, vol. cs.LG, Apr. 2010.

[20] K. Rawlik, M. Toussaint, and S. Vijayakumar, "Approximate Inference and Stochastic Optimal Control," *arXiv.org*, vol. cs.LG, Sept. 2010.

[21] G. Neumann, "Variational Inference for Policy Search in Changing Situations," in *International Conference on Machine Learning (ICML)*, 2011.

[22] Sutton, R.S. and Barto, A.G., *Reinforcement Learning: An Introduction*. Cambridge Univ Press, 1998.

[23] R. Sutton and D. Precup, "Between MDPs and Semi-MDPs: A Framework for Temporal Abstraction in Reinforcement Learning," *Artificial intelligence (AI)*, 1999.

[24] R. Sutton, "Generalization in Reinforcement Learning: Successful Examples Using Sparse Coarse Coding," *Advances in Neural Information Processing Systems (NIPS)*, 1996.

[25] S. Schaal, J. Peters, J. Nakanishi, and A. J. Ijspeert, "Learning Movement Primitives," in *International Symposium on Robotics Research (ISRR)*, 2003.

[26] "http://www.buylifetime.com/ products/ blt/ pid-90029.aspx."