Policy Gradient Methods for Robotics

Jan Peters*, Stefan Schaal*†
* *University of Southern California, Los Angeles, CA 90089, USA*
† *ATR Computational Neuroscience Laboratories, Kyoto, 619-0288, Japan*
{*jrpeters, sschaal*}*@usc.edu*

Abstract— **The aquisition and improvement of motor skills and control policies for robotics from trial and error is of essential importance if robots should ever leave precisely pre-structured environments. However, to date only few existing reinforcement learning methods have been scaled into the domains of high-dimensional robots such as manipulator, legged or humanoid robots. Policy gradient methods remain one of the few exceptions and have found a variety of applications. Nevertheless, the application of such methods is not without peril if done in an uninformed manner. In this paper, we give an overview on learning with policy gradient methods for robotics with a strong focus on recent advances in the field. We outline previous applications to robotics and show how the most recently developed methods can significantly improve learning performance. Finally, we evaluate our most promising algorithm in the application of hitting a baseball with an anthropomorphic arm.**

I. Introduction

In order to ever leave the well-structured environments of factory floors and research labs, future robots will require the ability to aquire novel behaviors, motor skills and control policies as well as to improve existing ones. Reinforcement learning is probably the most general framework in which such robot learning problems can be phrased. However, most of the methods proposed in the reinforcement learning community to date are not applicable to robotics as they do not scale beyond robots with more than one to three degrees of freedom. Policy gradient methods are a notable exception to this statement. Starting with the pioneering work of Gullapali, Franklin and Benbrahim [1], [2] in the early 1990s, these methods have been applied to a variety of robot learning problems ranging from simple control tasks (e.g., balancing a ball-on a beam [3], and pole-balancing [4]) to complex learning tasks involving many degrees of freedom such as learning of complex motor skills [2], [5], [6] and locomotion [7]–[14][1].

The advantages of policy gradient methods for robotics are numerous. Among the most important ones are that the policy representations can be chosen so that it is meaningful for the task and can incorporate previous domain knowledge, that often fewer parameters are needed in the learning process than in value-function based approaches and that there is a variety of different algorithms for policy gradient estimation in the literature which have a rather strong theoretical underpinning. Additionally, policy gradient methods can be used model-free and therefore also be applied to robot problems without an in-depth understanding of the problem or mechanics of the robot[2].

Nevertheless, many of the current papers which apply policy gradient methods in robotics neglect the current state of the art in policy gradients. Therefore most of the practical applications require smart tricks in order to be applicable and are often not safe for usage on real robots as infeasible policies might have to be generated. The main goal of this paper is to review which policy gradient methods are applicable to robotics and which issues matter. The remainder of this paper will proceed as follows: firstly, we will pose the general assumptions and the problem statement for this paper. Secondly, we will discuss the different approaches to policy gradient estimation and discuss their applicability to robot planning and control. We focus on the most useful methods and discuss several algorithms in-depth. The presented algorithms in this paper are highly optimized versions of the original algorithms – some of these are unpublished in these more efficient forms. Thirdly, we show how these methods can be applied to motor skill learning in robotics and show learning results with a seven degrees of freedom, anthropomorphic SARCOS Master ARM.

A. General Assumptions and Notations

Most robotics domains require the state space and the action spaces to be continuous such that learning methods based on discretizations are not applicable for higher dimensional systems. However, as the policy is usually implemented on a digital computer, we assume that we can model the control system in a discrete-time manner and we will denote the current time step by k. In order to take possible stochasticity of the plant into account, we denote it using a probability distribution $\mathbf{x}_{k+1} \sim p\left(\mathbf{x}_{k+1} \left| \mathbf{x}_k, \mathbf{u}_k \right.\right)$ as model where $\mathbf{u}_k \in \mathbb{R}^M$ denotes the current action, and $\mathbf{x}_k$, $\mathbf{x}_{k+1} \in \mathbb{R}^N$ denote the current and next state, respectively. We furthermore assume that actions are generated by a policy $\mathbf{u}_k \sim \pi_\theta\left(\mathbf{u}_k \left| \mathbf{x}_k \right.\right)$ which is modeled as a probability distribution in order to incorporate exploratory actions; for some special problems, the optimal solution to a control problem is actually a stochastic controller [17]. The policy is assumed to be parameterized by some policy parameters $\theta \in \mathbb{R}^K$.The sequence of states and actions

[1] Recent advances in robot learning [15] clearly indicate that policy gradient methods are currently the most feasible choice for robotics and that few other algorithms appear applicable. Some methods used in robotics estimate and apply policy gradients but are not well-known as such, e.g., differential dynamic programming usually estimates a model-based gradient and the PEGASUS [16] trick is often used to estimate finite difference gradients efficiently.

[2] Note that the focus in this paper is on reinforcement learning algorithms applicable to robotics and not on solely geared for robotics as such usually have just a limited application even in the actual field of robotics.

forms a trajectory (also called history or roll-out) denoted by $\tau = [\mathbf{x}_{0:H}, \mathbf{u}_{0:H}]$ where H denotes the horizon which can be infinite. At each instant of time, the learning system receives a reward denoted by $r(\mathbf{x}_k, \mathbf{u}_k) \in \mathbb{R}$.

B. Problem Statement

The general goal of policy optimization in reinforcement learning is to optimize the policy parameters $\theta \in \mathbb{R}^K$ so that the expected return

$$J(\theta) = E\left\{\sum\nolimits_{k=0}^{H} a_k r_k\right\} \tag{1}$$

is optimized where a_k denote time-step dependent weighting factors, often set to $a_k = \gamma^k$ for discounted reinforcement learning (where γ is in $[0, 1]$) or $a_k = 1/H$ for the average reward case. For robotics, we require that any change to the policy parameterization has to be smooth as drastic changes can be hazardous for the robot, and for its environnment as useful initializations of the policy based on domain knowledge would otherwise vanish after a single update step. For these reasons, policy gradient methods which follow the steepest descent on the expected return are the method of choice. These methods update the policy parameterization according to the gradient update rule

$$\theta_{h+1} = \theta_h + \alpha_h \left. \nabla_\theta J \right|_{\theta=\theta_h}, \tag{2}$$

where $\alpha_h \in \mathbb{R}^+$ denotes a learning rate and $h \in \{0, 1, 2, \ldots\}$ the current update number. If the gradient estimate is unbiased and learning rates fulfill $\sum_{h=0}^{\infty} \alpha_h > 0$ and $\sum_{h=0}^{\infty} \alpha_h^2 = \text{const}$, the learning process is guaranteed to converge to at least a local minimum.

II. Policy Gradient Methods for Robotics

The main problem in policy gradient methods is to obtain a good estimator of the policy gradient $\left. \nabla_\theta J \right|_{\theta=\theta_h}$. In robotics and control, people have traditionally used deterministic model-based methods for obtaining the gradient [18]–[20]. However, in order to become autonomous we cannot expect to be able to model every detail of the robot and environment. Therefore, we need to estimate the policy gradient simply from data generated during the execution of a task, i.e., without the need for a model. In this section, we will study different approaches and discuss which of these are useful in robotics.

A. General Approaches to Policy Gradient Estimation

The literature on policy gradient methods has yielded a variety of estimation methods over the last years. The most prominent approaches, which have been applied to robotics are finite-difference and likelihood ratio methods, more well-known as REINFORCE methods in reinforcement learning.

1) Finite-difference Methods: Finite-difference methods are among the oldest policy gradient approaches; they originated from the stochastic simulation community and are quite straightforward to understand. The policy parameterization is varied by small increments $\mathbf{\Delta}\theta_i$ and for each policy parameter variation $\theta_h + \mathbf{\Delta}\theta_i$ roll-outs are performed which generate estimates $\Delta\hat{J}_j \approx J(\theta_h + \mathbf{\Delta}\theta_i) - J_{\text{ref}}$ of the expected return. There are different ways of choosing the reference value J_{ref}, e.g. forward-difference estimators with $J_{\text{ref}} = J(\theta_h)$ and central-difference estimators with $J_{\text{ref}} = J(\theta_h - \mathbf{\Delta}\theta_i)$. The policy gradient estimate $\mathbf{g}_{\text{FD}} \approx \left. \nabla_\theta J \right|_{\theta=\theta_h}$ can be estimated by regression yielding

$$\mathbf{g}_{\text{FD}} = \left(\mathbf{\Delta\Theta}^T \mathbf{\Delta\Theta}\right)^{-1} \mathbf{\Delta\Theta}^T \mathbf{\Delta}\hat{\mathbf{J}}, \tag{3}$$

where $\mathbf{\Delta\Theta} = [\mathbf{\Delta}\theta_1, \ldots, \mathbf{\Delta}\theta_I]^T$ and $\mathbf{\Delta}\hat{\mathbf{J}} = [\Delta\hat{J}_1, \ldots, \Delta\hat{J}_I]^T$ denote the I samples. This approach can be highly efficient in simulation optimization of deterministic systems [21] or when a common history of random numbers [22] is being used (the later is known as PEGASUS in reinforcement learning [16]), and can get close to a convergence rate of $O\left(I^{-1/2}\right)$ [22]. However, when used on a real system, the uncertainities degrade the performance resulting in convergence rates ranging between $O\left(I^{-1/4}\right)$ to $O\left(I^{-2/5}\right)$ depending on the chosen reference value [22]. An implementation of this algorithm is shown in Table I.

Due to the simplicity of this approach, such methods have been successfully applied to robotics in numerous applications [6], [9], [11], [14]. However, the straightforward application to robotics is not without peril as the generation of the $\mathbf{\Delta}\theta_j$ requires proper knowledge on the system, as badly chosen $\mathbf{\Delta}\theta_j$ can destabilize the policy so that the system becomes instable and the gradient estimation process is prone to fail. Problems on robot control often require that each element of the vector $\mathbf{\Delta}\theta_j$ can have a different order of magnitude making the generation particularly difficult. Therefore, this approach can only applied in robotics under strict supervision of a robotics engineer.

2) Likelihood Ratio Methods / REINFORCE: Likelihood ratio methods are driven by an important different insight. Assume that trajectories τ are generated from a system by roll-outs, i.e., $\tau \sim p_\theta(\tau) = p(\tau|\theta)$ with rewards $r(\tau) = \sum_{k=0}^{H} a_k r_k$. In this case, the policy gradient can be estimated using the likelihood ratio (see e.g. [22], [23]) or REINFORCE [24] trick, i.e., by

$$\nabla_\theta J(\theta) = \int_{\mathbb{T}} \nabla_\theta p_\theta(\tau)\, r(\tau) d\tau \tag{4}$$

$$= E\left\{\nabla_\theta \log p_\theta(\tau)\, r(\tau)\right\}, \tag{5}$$

TABLE I

Finite difference gradient estimator.

```
input: policy parameterization $\theta_h$.
repeat
    generate policy variation $\mathbf{\Delta}\theta_1$.
    estimate $\hat{J}_j \approx J(\theta_h + \mathbf{\Delta}\theta_i) = \left\langle \sum_{k=0}^{H} a_k r_k \right\rangle$ from roll-out.
    estimate $\hat{J}_{\text{ref}}$, e.g., $\hat{J}_{\text{ref}} = J(\theta_h - \mathbf{\Delta}\theta_i)$ from roll-out.
    compute $\Delta\hat{J}_j \approx J(\theta_h + \mathbf{\Delta}\theta_i) - J_{\text{ref}}$.
    compute gradient $\mathbf{g}_{\text{FD}} = \left(\mathbf{\Delta\Theta}^T \mathbf{\Delta\Theta}\right)^{-1} \mathbf{\Delta\Theta}^T \mathbf{\Delta}\hat{\mathbf{J}}$.
until gradient estimate $\mathbf{g}_{\text{FD}}$ converged.
return: gradient estimate $\mathbf{g}_{\text{FD}}$.
```

as $\int_{\mathbb{T}} \nabla_\theta p_\theta(\tau)\, r(\tau) d\tau = \int_{\mathbb{T}} p_\theta(\tau)\, \nabla_\theta \log p_\theta(\tau)\, r(\tau) d\tau$. Importantly, the derivative $\nabla_\theta \log p_\theta(\tau)$ can be computed without knowleged of the generating distribution $p_\theta(\tau)$ as $p_\theta(\tau) = p(\mathbf{x}_0) \prod_{k=0}^{H} p(\mathbf{x}_{k+1} | \mathbf{x}_k, \mathbf{u}_k)\, \pi_\theta(\mathbf{u}_k | \mathbf{x}_k)$ implies that

$$\nabla_\theta \log p_\theta(\tau) = \sum\nolimits_{k=0}^{H} \nabla_\theta \log \pi_\theta(\mathbf{u}_k | \mathbf{x}_k), \qquad (6)$$

i.e., the derivatives through the control system do not have to be computed[3]. As $\int_{\mathbb{T}} \nabla_\theta p_\theta(\tau)\, d\tau = 0$, a constant baseline can be inserted resulting into the gradient estimator

$$\nabla_\theta J(\theta) = E\left\{ \nabla_\theta \log p_\theta(\tau) \left(r(\tau) - b \right) \right\}, \qquad (7)$$

where $b \in \mathbb{R}$ can be chosen arbitrarily [24] but usually with the goal to minimize the variance of the gradient estimator. Therefore, the general path likelihood ratio estimator or episodic REINFORCE gradient estimator is given by

$$\mathbf{g}_{\text{RF}} = \left\langle \left(\sum\nolimits_{k=0}^{H} \nabla_\theta \log \pi_\theta(\mathbf{u}_k | \mathbf{x}_k) \right) \left(\sum\nolimits_{l=0}^{H} a_l r_l - b \right) \right\rangle,$$

where $\langle \cdot \rangle$ denotes the average over trajectories [24]. This type of method is guaranteed to converge to the true gradient at the fastest theoretically possible pace of $O\left(I^{-1/2}\right)$ where I denotes the number of roll-outs [22] even if the data is generated from a highly stochastic system. An implementation of this algorithm will be shown in Table II together with the estimator for the optimal baseline.

Besides the theoretically faster convergence rate, likelihood ratio gradient methods have a variety of advantages in comparison to finite difference methods when applied to robotics. As the generation of policy parameter variations is no longer needed, the complicated control of these variables can no longer endanger the gradient estimation process. Furthermore, in practice, already a single roll-out can suffice for an unbiased gradient estimate [21], [25] viable for a good policy update step, thus reducing the amount of roll-outs needed. Finally, this approach has yielded the most real-world robotics results [1], [2], [5], [7], [10], [12], [13]. In the subsequent two sections, we will strive to explain and improve this type of gradient estimator.

TABLE II

General likelihood ratio policy gradient estimator "Episodic REINFORCE" with an optimal baseline.

```
input: policy parameterization $\theta_h$.
repeat
    perform a trial and obtain $\mathbf{x}_{0:H}, \mathbf{u}_{0:H}, r_{0:H}$
    for each gradient element $g_h$
        estimate optimal baseline
        $b^h = \frac{\left\langle \left( \sum_{k=0}^{H} \nabla_{\theta_h} \log \pi_\theta(\mathbf{u}_k | \mathbf{x}_k) \right)^2 \sum_{l=0}^{H} a_l r_l \right\rangle}{\left\langle \left( \sum_{k=0}^{H} \nabla_{\theta_h} \log \pi_\theta(\mathbf{u}_k | \mathbf{x}_k) \right)^2 \right\rangle}$
        estimate the gradient element
        $g_h = \left\langle \left( \sum_{k=0}^{H} \nabla_{\theta_h} \log \pi_\theta(\mathbf{u}_k | \mathbf{x}_k) \right) \left( \sum_{l=0}^{H} a_l r_l - b^h \right) \right\rangle.$
    end for.
until gradient estimate $\mathbf{g}_{\text{FD}} = [g_1, \ldots, g_h]$ converged.
return: gradient estimate $\mathbf{g}_{\text{FD}} = [g_1, \ldots, g_h]$.
```

B. 'Vanilla' Policy Gradient Approaches

Despite the fast asymptotic convergence speed of the gradient estimate, the variance of the likelihood-ratio gradient estimator can be problematic in practice. For this reason, we will discuss several advances in likelihood ratio policy gradient optimization, i.e., the policy gradient theorem/GPOMDP and optimal baselines[4].

1) Policy gradient theorem/GPOMDP: The trivial observation that future actions do not depend on past rewards (unless the policy has been changed) can result in a significant reduction of the variance of the policy gradient estimate. This insight can be formalized as $E\left\{ \nabla_\theta \log \pi_\theta(\mathbf{u}_l | \mathbf{x}_l)\, r_k \right\} = 0$ for $l > k$ which is straightforward to verify. This allows two variations of the previous algorithm which are known as the policy gradient theorem [17]

$$\mathbf{g}_{\text{PGT}} = \left\langle \sum\nolimits_{k=0}^{H} \nabla_\theta \log \pi_\theta(\mathbf{u}_k | \mathbf{x}_k) \left(\sum\nolimits_{l=k}^{H} a_l r_l - b_k \right) \right\rangle,$$

or G(PO)MD [25]

$$\mathbf{g}_{\text{GMDP}} = \left\langle \sum\nolimits_{l=0}^{H} \left(\sum\nolimits_{k=0}^{l} \nabla_\theta \log \pi_\theta(\mathbf{u}_k | \mathbf{x}_k) \right) (a_l r_l - b_l) \right\rangle.$$

While these algorithms *look* different, they are *exactly equivalent* in their gradient estimate[5], i.e., $\mathbf{g}_{\text{PGT}} = \mathbf{g}_{\text{GMPD}}$, and have been derived previously in the simulation optimization community [27]. An implementation of this algorithm is shown together with the optimal baseline in Table III.

However, in order to clarify the relationship to [17], [25], we note that the term $\sum_{l=k}^{H} a_l r_l$ in the policy gradient theorem is equivalent to a monte-carlo estimate of the value function $Q_{k:H}^{\theta}(\mathbf{x}_k, \mathbf{u}_k) = E\left\{ \sum_{l=k}^{H} a_l r_l \middle| \mathbf{x}_k, \mathbf{u}_k \right\}$ and the term $\sum_{k=0}^{l} \nabla_\theta \log \pi_\theta(\mathbf{u}_k | \mathbf{x}_k)$ becomes the log-derivative of the distribution of states $\mu_\theta^k(\mathbf{x}_k)$ at step k in expectation, i.e., $\nabla_\theta \log \mu_\theta(\mathbf{x}_k) = E\left\{ \sum_{k=0}^{k} \nabla_\theta \log \pi_\theta(\mathbf{u}_k | \mathbf{x}_k) \middle| \mathbf{x}_k \right\}$. When either of these two constructs can be easily obtained by derivation or estimation, the variance of the gradient can be reduced significantly.

Without a formal derivation of it, the policy gradient theorem has been applied in robotics using estimated value functions $Q_{k:H}^{\theta}(\mathbf{x}_k, \mathbf{u}_k)$ instead of the term $\sum_{l=k}^{H} a_l r_l$ and a baseline $b_k = V_{k:H}^{\theta}(\mathbf{x}_k) = E\left\{ \sum_{l=k}^{H} a_l r_l \middle| \mathbf{x}_k \right\}$ [7], [28].

2) Optimal Baselines: Above, we have already introduced the concept of a baseline which can decrease the variance of a policy gradient estimate by orders of magnitude. Thus, an

[3] This result makes an important difference: in stochastic system optimization, finite difference estimators are often prefered as the derivative through system is required but not known. In policy search, we always know the derivative of the policy with respect to its parameters and therefore we can make use of the theoretical advantages of likelihood ratio gradient estimators.

[4] Note that the theory of the compatible function approximation [17] is omitted at this point as it does not contribute to practical algorithms in this context. For a thorough discussion of this topic see [5], [26].

[5] Note that [25] additionally add an eligibility trick for reweighting trajectory pieces. This trick can be highly dangerous in robotics as can be demonstrated that already in linear-quadratic regulation, this trick can result into divergence as the optimial policy for small planning horizons (i.e., small eligibility rates) is often instable.

optimal selection of such a baseline is essential. An optimal baseline minimizes the variance $\sigma_h^2 = \mathrm{Var}\{g_h\}$ of each element g_h of the gradient $\mathbf{g}$ *without* biasing the gradient estimate, i.e., violating $E\{\mathbf{g}\} = \nabla_\theta J$. This can be phrased as having a seperate baseline b^h for every element of the gradient[6]. Due to the requirement of unbiasedness of the gradient estimate, we have $\sigma_h^2 = E\left\{g_h^2\right\} - \left(\nabla_{\theta_h} J\right)^2$ and due to $\min_{b_h} \sigma_h^2 \geq E\left\{\min_{b_h} g_h^2\right\} - \left(\nabla_{\theta_h} J\right)^2$, the optimal baseline for each gradient element g_h can always be given by

$$b^h = \frac{\left\langle \left(\sum_{k=0}^{H} \nabla_{\theta_h} \log \pi_\theta\left(\mathbf{u}_k \left|\mathbf{x}_k\right.\right)\right)^2 \sum_{l=0}^{H} a_l r_l \right\rangle}{\left\langle \left(\sum_{k=0}^{H} \nabla_{\theta_h} \log \pi_\theta\left(\mathbf{u}_k \left|\mathbf{x}_k\right.\right)\right)^2 \right\rangle}$$

for the general likelihood ratio gradient estimator, i.e., Episodic REINFORCE. The algorithmic form of the optimal baseline is shown in Table II in line 4. If the sums in the baselines are modified appropriately, we can obtain the optimal baseline for the policy gradient theorem or G(PO)MPD. We only show G(PO)MDP in this paper in Table III as the policy gradient theorem is numerically equivalent.

The optimal baseline which does not bias the gradient in Episodic REINFORCE can only be a single number for all trajectories and in G(PO)MPD it can also depend on the time-step [35]. However, in the policy gradient theorem it can depend on the current state and, therefore, if a good parameterization for the baseline is known, e.g., in a generalized linear form $b\left(\mathbf{x}_k\right) = \phi\left(\mathbf{x}_k\right)^T \omega$, this can significantly improve the gradient estimation process. However, the selection of the basis functions $\phi\left(\mathbf{x}_k\right)$ can be difficult and often impractical in robotics. See [24], [29]–[34] for more information on this topic.

TABLE III

SPECIALIZED LIKELIHOOD RATIO POLICY GRADIENT ESTIMATOR "G(PO)MDP"/POLICY GRADIENT WITH AN OPTIMAL BASELINE.

```
input: policy parameterization $\theta_h$.
repeat
    perform trials and obtain $\mathbf{x}_{0:H}, \mathbf{u}_{0:H}, r_{0:H}$
    for each gradient element $g_h$
        for each time step $k$
            estimate baseline for time step $k$ by
            $b_k^h = \frac{\left\langle \left(\sum_{\kappa=0}^{k} \nabla_{\theta_h} \log \pi_\theta(\mathbf{u}_\kappa|\mathbf{x}_\kappa)\right)^2 a_k r_k \right\rangle}{\left\langle \left(\sum_{\kappa=0}^{k} \nabla_{\theta_h} \log \pi_\theta(\mathbf{u}_\kappa|\mathbf{x}_\kappa)\right)^2 \right\rangle}$
        end for.
        estimate the gradient element
        $g_h = \left\langle \sum_{l=0}^{H} \left(\sum_{k=0}^{l} \nabla_{\theta_h} \log \pi_\theta\left(\mathbf{u}_k \left|\mathbf{x}_k\right.\right)\right) \left(a_l r_l - b_l^h\right) \right\rangle.$
    end for.
until gradient estimate $\mathbf{g}_{\mathrm{FD}} = [g_1, \ldots, g_h]$ converged.
return: gradient estimate $\mathbf{g}_{\mathrm{FD}} = [g_1, \ldots, g_h]$.
```

C. Natural Actor-Critic Approaches

One of the main reasons for using policy gradient methods is that we intend to do just a small change $\boldsymbol{\Delta}\theta$ to the policy π_θ while improving the policy. However, the meaning of small is ambiguous. When using the Euclidian metric of $\sqrt{\boldsymbol{\Delta}\theta^T \boldsymbol{\Delta}\theta}$, then the gradient is different for every parameterization θ of the policy π_θ even if these parameterization are related to each other by a linear transformation [36]. This poses the question of how we can measure the closeness between the current policy and the updated policy based upon the distribution of the paths generated by each of these. In statistics, a variety of distance measures for the closeness of two distributions (e.g., $p_\theta\left(\tau\right)$ and $p_{\theta+\boldsymbol{\Delta}\theta}\left(\tau\right)$) have been suggested, e.g., the Kullback-Leibler divergence[7] $d_{\mathrm{KL}}\left(p_\theta, p_{\theta+\boldsymbol{\Delta}\theta}\right)$, the Hellinger distance d_{HD} and others [38]. Many of these distances (e.g., the previously mentioned ones) can be approximated by the same second order Taylor expansion, i.e., by

$$d_{\mathrm{KL}}\left(p_\theta, p_{\theta+\boldsymbol{\Delta}\theta}\right) \approx \boldsymbol{\Delta}\theta^T \mathbf{F}_\theta \ \boldsymbol{\Delta}\theta, \tag{8}$$

where $\mathbf{F}_\theta = \int_{\mathbb{T}} p_\theta\left(\tau\right) \nabla \log p_\theta\left(\tau\right) \nabla \log p_\theta\left(\tau\right)^T d\tau = \left\langle \nabla \log p_\theta\left(\tau\right) \nabla \log p_\theta\left(\tau\right)^T \right\rangle$ is known as the Fisher-information matrix. Let us now assume that we restrict the change of our policy to the length of our step-size α_n, i.e., we have a restricted step-size gradient descent approach well-known in the optimization literature [39], and given by

$$\boldsymbol{\Delta}\theta = \mathrm{argmax}_{\boldsymbol{\Delta}\tilde{\theta}} \frac{\alpha_n \boldsymbol{\Delta}\tilde{\theta}^T \nabla_\theta J}{\boldsymbol{\Delta}\tilde{\theta}^T \mathbf{F}_\theta \ \boldsymbol{\Delta}\tilde{\theta}} = \alpha_n \mathbf{F}_\theta^{-1} \nabla_\theta J, \tag{9}$$

where $\nabla_\theta J$ denotes the 'vanilla' policy gradient from Section II-B. This update step can be interpreted as follows: determine the maximal improvement $\boldsymbol{\Delta}\tilde{\theta}$ of the policy for a constant fixed change of the policy $\boldsymbol{\Delta}\tilde{\theta}^T \mathbf{F}_\theta \boldsymbol{\Delta}\tilde{\theta}$.

This type of approach is known as Natural Policy Gradients and has its separate origin in supervised learning [40]. It was first suggested in the context of reinforcement learning by Kakade [36] and has been explored in greater depth in [5], [26], [35], [41]. The strongest theoretical advantage of this approach is that its performance no longer depends on the parameterization of the policy and it is therefore safe to use for arbitrary policies[8]. In practice, the learning process converges significantly faster in most practical cases.

1) Episodic Natural Actor-Critic: One of the fastest general algorithms for estimating natural policy gradients which does not need complex parameterized baselines is the episodic natural actor critic. This algorithm, originally derived in [5], [26], [35], can be considered the 'natural' version of reinforce with a baseline optimal for this gradient estimator. However, for steepest descent with respect to a metric, the baseline also needs to minimize the variance with respect to the same metric. In this case, we can minimize the whole covariance matrix of

[6] A single baseline for all parameters can also be obtained and is more common in the reinforcement learning literature [24], [29]–[34]. However, such a baseline is of course suboptimal.

[7] While being 'the natural way to think about closeness in probability distributions' [37], this measure is technically not a metric as it is not commutative.

[8] There is a variety of interesting properties to the natural policy gradient methods which are explored in [5].

the natural gradient estimate $\boldsymbol{\Delta}\hat{\theta}$ given by

$$\begin{aligned}\boldsymbol{\Sigma} &= \mathrm{Cov}\left\{\boldsymbol{\Delta}\hat{\theta}\right\}_{\mathbf{F}_\theta} \\ &= E\left\{\left(\boldsymbol{\Delta}\hat{\theta}-\mathbf{F}_\theta^{-1}\mathbf{g}_{\mathrm{LR}}\left(b\right)\right)^T\mathbf{F}_\theta\left(\boldsymbol{\Delta}\hat{\theta}-\mathbf{F}_\theta^{-1}\mathbf{g}_{\mathrm{LR}}\left(b\right)\right)\right\},\end{aligned}$$

with $\mathbf{g}_{\mathrm{LR}}\left(b\right) = \left\langle\nabla\log p_\theta\left(\tau\right)\left(r\left(\tau\right)-b\right)\right\rangle$ being the REINFORCE gradient with baseline b. As outlined in [5], [26], [35], it can be shown that the minimum-variance unbiased natural gradient estimator can be determined as shown in Table IV.

2) Episodic Natural Actor Critic with a Time-Variant Baseline: The episodic natural actor critic described in the previous section suffers from drawback: it does not make use of intermediate data just like REINFORCE. For policy gradients, the way out was G(PO)MDP which left out terms which would average out in expectation. In the same manner, we can make the argument for a time-dependent baseline which then allows us to reformulate the Episodic Natural Actor Critic. This results in the algorithm shown in Table V. The advantage of this type of algorithms is two-fold: the variance of the gradient estimate is often lower and it can take time-variant rewards significantly better into account.

III. Experiments & Results

In the previous section, we outlined the five first-order, model-free policy gradient algorithms which are most relevant for robotics (further ones exist but are do not scale into high-dimensional robot domains). In this section, we will demonstrate how these different algorithms compare in practice in different areas relevant to robotics. For this pupose, we will show experiments on both simulated plants as well as on real robots and we will compare the algorithms for the optimization of control laws and for learning of motor skills.

A. Control Law Optimization

As a first comparison, we use the linearized cartpole standard example which is described in detail in [26] (however, as [26] does not focus on episodic tasks, we cannot compare these results). We have a cart-pole system simulated as a linear stochastic system $p(\mathbf{x}'|\mathbf{x},\mathbf{u}) = \mathcal{N}(\mathbf{x}'|\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{u},\hat{\sigma}^2\mathbf{I})$ around its equilibrium point (up-right pole and centered cart with zero verlocities) and using a quadratic reward $r(\mathbf{x},\mathbf{u}) = \mathbf{x}^T\mathbf{Q}\mathbf{x}+\mathbf{u}^T\mathbf{R}\mathbf{u}$. Each system is given a stochastic policy $\pi(u|\mathbf{x}) = \mathcal{N}(u|\theta_{\text{gain}}^T\mathbf{x},\sigma_{\text{base}}+\exp(\theta_{\text{exploration}}))$ with policy parameters $\theta = [\theta_{\text{gain}}^T,\theta_{\text{exploration}}]^T$. As the optimal solution is well-known in this example and can be derived using dynamic programming, this serves as good benchmark. The larges stochasticity injected into the system makes this a hard problem. The learning system is supposed to optimize the policy without knowledge of the system. The results shown in Figure 1 illustrate the efficiency of natural gradients:

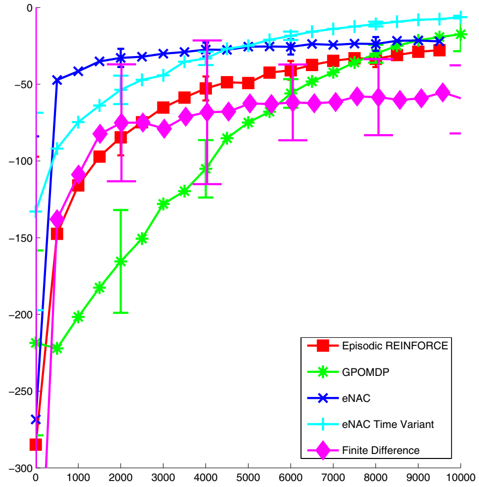


Fig. 1. This figure shows a comparison of the algorithms presented in this paper on a linearized cart-pole standard benchmark.

TABLE IV

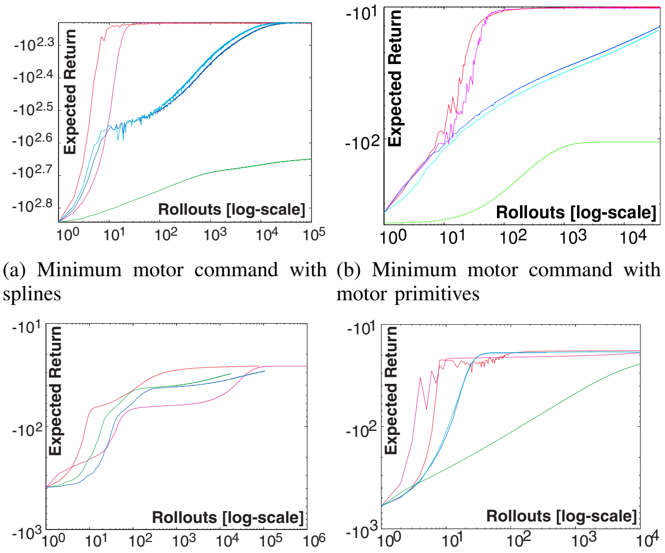
Episodic Natural Actor Critic

```
input: policy parameterization $\theta_h$.
repeat
    perform $M$ trials and obtain $\mathbf{x}_{0:H},\mathbf{u}_{0:H},r_{0:H}$ for each trial.
    Obtain the sufficient statistics
        Policy derivatives $\psi_k = \nabla_\theta\log\pi_\theta\left(\mathbf{u}_k\left|\mathbf{x}_k\right.\right)$.
        Fisher matrix $\mathbf{F}_\theta = \left\langle\left(\sum_{k=0}^H\psi_k\right)\left(\sum_{l=0}^H\psi_l\right)^T\right\rangle$.
        Vanilla gradient $\mathbf{g} = \left\langle\left(\sum_{k=0}^H\psi_k\right)\left(\sum_{l=0}^H a_l r_l\right)\right\rangle$.
        Eligbility $\phi = \left\langle\left(\sum_{k=0}^H\psi_k\right)\right\rangle$.
        Average reward $\bar{r} = \left\langle\sum_{l=0}^H a_l r_l\right\rangle$.
    Obtain natural gradient by computing
        Baseline $b = \mathbf{Q}\left(\bar{r}-\phi^T\mathbf{F}_\theta^{-1}\mathbf{g}\right)$
        with $\mathbf{Q} = M^{-1}\left(1+\phi^T\left(M\mathbf{F}_\theta-\phi\phi^T\right)^{-1}\phi\right)$
        Natural gradient $\mathbf{g}_{\mathrm{NG}} = \mathbf{F}_\theta^{-1}\left(\mathbf{g}-\phi b\right)$.
until gradient estimate $\mathbf{g}_{\mathrm{NG}} = [g_1,\ldots,g_h]$ converged.
return: gradient estimate $\mathbf{g}_{\mathrm{NG}} = [g_1,\ldots,g_h]$.
```

TABLE V

Episodic Natural Actor Critic with a Time-Variant Baseline

```
input: policy parameterization $\theta_h$.
repeat
    perform $M$ trials and obtain $\mathbf{x}_{0:H},\mathbf{u}_{0:H},r_{0:H}$ for each trial.
    Obtain the sufficient statistics
        Policy derivatives $\psi_k = \nabla_\theta\log\pi_\theta\left(\mathbf{u}_k\left|\mathbf{x}_k\right.\right)$.
        Fisher matrix $\mathbf{F}_\theta = \left\langle\sum_{k=0}^H\left(\sum_{l=0}^k\psi_l\right)\psi_k^T\right\rangle$.
        Vanilla gradient $\mathbf{g} = \left\langle\sum_{k=0}^H\left(\sum_{l=0}^k\psi_l\right)a_k r_k\right\rangle$,
        Eligbility matrix $\mathbf{\Phi} = [\phi_1,\phi_2,\ldots,\phi_K]$
        with $\phi_h = \left\langle\left(\sum_{k=0}^h\psi_k\right)\right\rangle$.
        Average reward vector $\bar{\mathbf{r}} = [\bar{r}_1,\bar{r}_2,\ldots,\bar{r}_K]$
        with $\bar{r}_h = \langle a_h r_h\rangle$.
    Obtain natural gradient by computing
        Baseline $\mathbf{b} = \mathbf{Q}\left(\bar{\mathbf{r}}-\mathbf{\Phi}^T\mathbf{F}_\theta^{-1}\mathbf{g}\right)$
        with $\mathbf{Q} = M^{-1}\left(\mathbf{I}_K+\mathbf{\Phi}^T\left(M\mathbf{F}_\theta-\mathbf{\Phi}\mathbf{\Phi}^T\right)^{-1}\mathbf{\Phi}\right)$.
        Natural gradient $\mathbf{g}_{\mathrm{NG}} = \mathbf{F}_\theta^{-1}\left(\mathbf{g}-\mathbf{\Phi}\mathbf{b}\right)$.
until gradient estimate $\mathbf{g}_{\mathrm{NG}} = [g_1,\ldots,g_h]$ converged.
return: gradient estimate $\mathbf{g}_{\mathrm{NG}} = [g_1,\ldots,g_h]$.
```



(a) Minimum motor command with splines

(b) Minimum motor command with motor primitives

(c) Passing through a point with splines

(d) Passing through a point with motor primitives

Fig. 2. This figure shows different experiments with motor task learning. In (a,b), we see how the learning system creates minimum motor command goal-achieving plans using both (a) splines and (b) motor primitives. For this problem, the natural actor-critic methods beat all other methods by several orders of magnitude. In (c,d), the plan has to achieve an intermediary goal. While the natural actor-critic methods still outperform previous methods, the gap is lower as the learning problem is easier. Note that these are double logarithmic plots.

Episodic REINFORCE is by an order of magnitude slower than the Episodic Natural Actor-Critic; the same holds true on G(PO)MDP and the time-variant baseline version of the Episodic Natural Actor-Critic. The lower variance of the time-variant versions allows a higher learning rate, hence the faster convergence. The finite-difference gradient estimator performs initially well although with a large variance. However, closer to the optimal solution, the performance of the gradient estimate degraded due to the increased influence of the stochasticty of the system. In general, the episodic natural actor-critic with a time-variant baseline performs best.

B. Motor Task Planning

This section will turn towards optimizing motor plans for robotics. For this purpose, we consider two forms of motor plans, i.e., (1) *spline-based trajectory plans* and (2)*nonlinear dynamic motor primitives* introduced in [42]. Spline-based trajectory planning is well-known in the robotics literature, see e.g., [43], [44]. A desired trajectory is represented by piecewise connected polynomials, i.e., we have $y_i\left(t\right) = \theta_{0i} + \theta_{1i}t + \theta_{2i}t^2 + \theta_{3i}t^3$ in $t \in [t_i, t_{i+1}]$ under the constraints that both $y_i\left(t_{i+1}\right) = y_{i+1}\left(t_{i+1}\right)$ and $\dot{y}_i\left(t_{i+1}\right) = \dot{y}_{i+1}\left(t_{i+1}\right)$. A PD control law ensures that the trajectory is tracked well. For nonlinear dynamic motor primitives, we use the approach developed in [42] where movement plans $(\mathbf{q}_d, \dot{\mathbf{q}}_d)$ for each degree of freedom (DOF) of the robot is represented in terms of the time evolution of the nonlinear dynamical systems

$$\ddot{q}_{d,k} = h(q_{d,k}, \mathbf{z}_k, g_k, \tau, \theta_k) \tag{10}$$

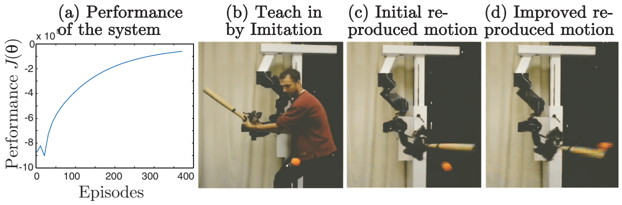
where $(q_{d,k}, \dot{q}_{d,k})$ denote the desired position and velocity of a joint, z_k the internal state of the dynamic system, g_k the goal (or point attractor) state of each DOF, τ the movement duration shared by all DOFs, and θ_k the open parameters of the function h. The original work in [42] demonstrated how the parameters θ_k can be learned to match a template trajectory by means of supervised learning – this scenario is, for instance, useful as the first step of an imitation learning system. Here we will add the ability of self-improvement of the movement primitives in Eq.(10) by means of reinforcement learning, which is the crucial second step in imitation learning. The system in Eq.(10) is a point-to-point movement, i.e., this task is rather well suited for the introduced episodic reinforcement learning methods. In Figure 2 (a) and (b), we show a comparison of the presented algorithms for a simple, single DOF task with a reward of $r_k(x_{0:N}, u_{0:N}) = \sum_{i=0}^{N} c_1 \dot{q}_{d,k,i}^2 + c_2(q_{d;k;N} - g_k)^2$; where $c_1 = 1$, $c_2 = 1000$ for both splines and dynamic motor primitives. In Figure 2 (c) and (d) we show the same with an additional punishment term for going through a intermediate point p_F at time F, i.e., $r_k(x_{0:N}, u_{0:N}) = \sum_{i=0}^{N} \tilde{c}_1 \dot{q}_{d,k,i}^2 + \tilde{c}_2(q_{d;k;N} - g_k)^2 + \tilde{c}_2(q_{d;F;N} - p_F)^2$. It is quite clear from the results that the natural actor-critic methods outperform both the vanilla policy gradient methods as well as the likelihood ratio methods. Finite difference gradient methods behave differently from the previous experiment as there is no stochasticity in the system, resulting in a cleaner gradient but also in local minima not present for likelihood ratio methods where the exploratory actions are stochastic.

C. Motor Primitive Learning for Baseball

We also evaluated the same setup in a challenging robot task, i.e., the planning of these motor primitives for a seven DOF robot task using our SARCOS Master Arm. The task of the robot is to hit the ball properly so that it flies as far as possible; this game is also known as T-Ball. The state of the robot is given by its joint angles and velocities while the action are the joint accelerations. The reward is extracted using color segment tracking with a NewtonLabs vision system. Initially, we teach a rudimentary stroke by supervised learning as can be seen in Figure 3 (b); however, it fails to reproduce the behavior as shown in (c); subsequently, we improve the performance using the episodic Natural Actor-Critic which yields the performance shown in (a) and the behavior in (d). After approximately 200-300 trials, the ball can be hit properly by the robot.

IV. Conclusion

We have presented an extensive survey of policy gradient methods. While some developments needed to be omitted as they are only applicable for very low-dimensional state-spaces, this paper represents the state of the art in policy gradient methods and can deliver a solid base for future applications of policy gradient methods in robotics. All three major ways of estimating first order gradients, i.e., finite-difference gradients, vanilla policy gradients and natural policy gradients are discussed in this paper and practical algorithms



(a) Performance of the system (b) Teach in by Imitation (c) Initial reproduced motion (d) Improved reproduced motion

Fig. 3. This figure shows (a) the performance of a baseball swing task when using the motor primitives for learning. In (b), the learning system is initialized by imitation learning, in (c) it is initially failing at reproducing the motor behavior, and (d) after several hundred episodes exhibiting a nicely learned batting.

are given. The experiments presented here show that the time-variant episodic natural actor critic is the preferred method when applicable; however, if a policy cannot be differentiated with respect to its parameters, the finite difference methods may be the only method applicable. The example of motor primitive learning for baseball underlines the efficiency of natural gradient methods.

References

[1] H. Benbrahim and J. Franklin, "Biped dynamic walking using reinforcement learning," *Robotics and Autonomous Systems*, vol. 22, pp. 283–302, 1997.

[2] V. Gullapalli, J. Franklin, and H. Benbrahim, "Aquiring robot skills via reinforcement learning," *IEEE Control Systems*, vol. -, no. 39, 1994.

[3] H. Benbrahim, J. Doleac, J. Franklin, and O. Selfridge, "Real-time learning: A ball on a beam," in *Proceedings of the 1992 International Joint Conference on Neural Networks*, Baltimore, MD, 1992.

[4] H. Kimura and S. Kobayashi, "Reinforcement learning for continuous action using stochastic gradient ascent," in *The 5th International Conference on Intelligent Autonomous Systems*, 1998.

[5] J. Peters, S. Vijayakumar, and S. Schaal, "Natural actor-critic," in *Proceedings of the European Machine Learning Conference (ECML)*, 2005.

[6] N. Mitsunaga, C. Smith, T. Kanda, H. Ishiguro, and N. Hagita, "Robot behavior adaptation for human-robot interaction based on policy gradient reinforcement learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*, 2005, pp. 1594–1601.

[7] H. Kimura and S. Kobayashi, "Reinforcement learning for locomotion of a two-linked robot arm," *Proceedings of the 6th Europian Workshop on Learning Robots EWLR-6*, pp. 144–153, 1997.

[8] M. Sato, Y. Nakamura, and S. Ishii, "Reinforcement learning for biped locomotion," in *International Conference on Artificial Neural Networks (ICANN)*, ser. Lecture Notes in Computer Science. Springer-Verlag, 2002, pp. 777–782.

[9] N. Kohl and P. Stone, "Policy gradient reinforcement learning for fast quadrupedal locomotion," in *Proceedings of the IEEE International Conference on Robotics and Automation*, New Orleans, LA, May 2004.

[10] G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng, "Learning cpg sensory feedback with policy gradient for biped locomotion for a full-body humanoid," in *AAAI 2005*, 2005.

[11] R. Tedrake, T. W. Zhang, and H. S. Seung, "Learning to walk in 20 minutes," in *Proceedings of the Fourteenth Yale Workshop on Adaptive and Learning Systems*, 2005.

[12] T. Mori, Y. Nakamura, M. Sato, and S. Ishii, "Reinforcement learning for cpg-driven biped robot," in *AAAI 2004*, 2004, pp. 623–630.

[13] S. I. Yutaka Nakamura, Takeshi Mori, "Natural policy gradient reinforcement learning for a cpg control of a biped robot," in *PPSN 2004*, 2004, pp. 972–981.

[14] T. Geng, B. Porr, and F. Wörgötter, "Fast biped walking with a reflexive neuronal controller and real-time online learning," *Submitted to Int. Journal of Robotics Res.*, 2005.

[15] J. Peters, S. Schaal, and R. Tedrake, "Workshop on learning for locomotion," in *Robotics: Science & Systems*, 2005.

[16] A. Y. Ng and M. Jordan, "Pegasus: A policy search method for large mdps and pomdps," in *Uncertainty in Artificial Intelligence, Proceedings of the Sixteenth Conference*, 2000.

[17] R. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," *Advances in Neural Information Processing Systems*, vol. 12, no. 22, 2000.

[18] D. H. Jacobson and D. Q. Mayne, *Differential Dynamic Programming*. New York: American Elsevier Publishing Company, Inc., 1970.

[19] P. Dyer and S. R. McReynolds, *The Computation and Theory of Optimal Control*. New York: Academic Press, 1970.

[20] L. Hasdorff, *Gradient optimization and nonlinear control*. John Wiley & Sons, 1976.

[21] J. C. Spall, *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. Hoboken, NJ: Wiley, 2003.

[22] P. Glynn, "Likelihood ratio gradient estimation: an overview," in *Proceedings of the 1987 Winter Simulation Conference*, Atlanta, GA, 1987, pp. 366–375.

[23] V. Aleksandrov, V. Sysoyev, and V. Shemeneva, "Stochastic optimization," *Engineering Cybernetics*, vol. 5, pp. 11–16, 1968.

[24] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, no. 23, 1992.

[25] J. Baxter and P. Bartlett, "Direct gradient-based reinforcement learning," *Journal of Artificial Intelligence Research*, 1999. [Online]. Available: citeseer.nj.nec.com/baxter99direct.html

[26] J. Peters, S. Vijayakumar, and S. Schaal, "Reinforcement learning for humanoid robotics," in *IEEE/RSJ International Conference on Humanoid Robotics*, 2003.

[27] P. Glynn, "Likelihood ratio gradient estimation for stochastic systems," *Communications of the ACM*, vol. 33, no. 10, pp. 75–84, October 1990.

[28] V. Gullapalli, "Associative reinforcement learning of real-value functions," *SMC*, vol. -, no. -, 1991.

[29] L. Weaver and N. Tao, "The optimal reward baseline for gradient-based reinforcement learning," *Uncertainty in Artificial Intelligence: Proceedings of the Seventeenth Conference*, vol. 17, no. 29, 2001.

[30] E. Greensmith, P. L. Bartlett, and J. Baxter, "Variance reduction techniques for gradient estimates in reinforcement learning," *Journal of Machine Learning Research*, vol. 5, pp. 1471–1530, Nov. 2004.

[31] L. Weaver and N. Tao, "The variance minimizing constant reward baseline for gradient-based reinforcement learning," *Technical Report ANU*, vol. -, no. 30, 2001.

[32] G. Lawrence, N. Cowan, and S. Russell, "Efficient gradient estimation for motor control learning," in *Proc. UAI-03*, Acapulco, Mexico, 2003.

[33] A. Berny, "Statistical machine learning and combinatorial optimization," *In L. Kallel, B. Naudts, and A. Rogers, editors, Theoretical Aspects of Evolutionary Computing, Lecture Notes in Natural Computing*, vol. 0, no. 33, 2000.

[34] E. Greensmith, P. Bartlett, and J. Baxter, "Variance reduction techniques for gradient estimates in reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 14, no. 34, 2001.

[35] J. Peters, "Machine learning of motor skills for robotics," Ph.D. Thesis Proposal / USC Technical Report, Tech. Rep., 2005.

[36] S. Kakade, "A natural policy gradient," in *Advances in Neural Information Processing Systems*, vol. 14, no. 26, 2001.

[37] V. Balasubramanian, "Statistical inference, occam's razor, and statistical mechanics on the space of probability distributions," *Neural Computation*, vol. 9, no. 2, pp. 349–368, 1997.

[38] F. Su and A. Gibbs, "On choosing and bounding probability metrics," *International Statistical Review*, vol. 70, no. 3, pp. 419–435, 2002.

[39] R. Fletcher and R. Fletcher, *Practical Methods of Optimization*. John Wiley & Sons, 2000.

[40] S. Amari, "Natural gradient works efficiently in learning," *Neural Computation*, vol. 10, 1998.

[41] J. Bagnell and J. Schneider, "Covariant policy search," *International Joint article on Artificial Intelligence*, 2003.

[42] A. Ijspeert, J. Nakanishi, and S. Schaal, "Learning rhythmic movements by demonstration using nonlinear oscillators," pp. 958–963, 2002.

[43] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators*. Springer Verlag, 2000.

[44] H. Miyamoto, S. Schaal, F. Gandolfo, Y. Koike, R. Osu, E. Nakano, Y. Wada, and M. Kawato, "A kendama learning robot based on bi-directional theory," *Neural Networks*, vol. 9, no. 8, pp. 1281–1302, 1996.