Universität Stuttgart

Universität Stuttgart
Fakultät 7: Konstruktions-, Produktions- und Fahrzeugtechnik
Institut für Angewandte und Experimentelle Mechanik
Prof. Dr.-Ing. Arnold Kistner

Max-Planck-Institut für biologische Kybernetik Tübingen
Empirische Inferenz
Prof. Dr. Bernhard Schölkopf

DIPLOMARBEIT

Reinforcement Learning for Motor Primitives

Vorgelegt von: Jens Kober
geboren am: 6. Mai 1982 in: Künzelsau

zum
Erlangen des akademischen Grades

DIPLOM-INGENIEUR
(Dipl.-Ing.)

Betreuer: Dr./USC Jan Peters, Leiter des MPI Robot Learning Lab

Aims and Objectives

Title:

Reinforcement Learning for Motor Primitives

Author:

Jens Kober, Cand.-Ing.

Advisor:

Jan Peters, Dr./USC, Dipl.-Ing., Dipl.-Inf., M.Sc. (CS), M.Sc. (AME)

Abstract:

Humans demonstrate a large variety of very complicated motor skills in their day-to-day life. Their agility and adaptability to new control task remains unmatched by the millions of robots laboring on factory floors and roaming research labs. Achieving the abilities of learning and improving new motor skills has become an essential component in order to get a step closer to human-like motor skills. If future robots could acquire their basic task by imitating human demonstrations and subsequently self-improve by trial and error, such robot learning would result into more interesting robot applications as well as large productivity gains in industry.

Recent progress in the area of machine learning has yielded several important tools for making progress towards this vision for the future. Two of these recent developments are a novel framework for representing motor primitives using dynamical systems presented in [Ijspeert et al., 2002a,b, 2003, Schaal et al., 2004] and the reduction of reward-related self-improvement to reward-weighted regression by Peters and Schaal [2006c, 2007b]. To date, these two important methods have only been used separately in robot learning; when using them in combination, this could yield a strong basis for learning motor skills.

In this diploma thesis, the task of ball-in-a-cup or balero was chosen as a target evaluation platform for the proposed method. We will start by first implementing a simulation platform for the balero task in the physically realistic robot simulator SL [Schaal, 2004]. Subsequently, we obtain human presentations of the task using a VICON^{TM} motion capture system. This data is used for imitations learning and we will show that several very different basic movements exist which can fulfill the task. We implement the framework of motor primitives based on dynamical systems, adapt it for applicability to our task and subsequently discuss how the suggested learning framework works in toy applications. Using the physically realistic simulation, we evaluate our resulting framework in simulation and test it extensively. Final evaluations are planned to take place on a high-speed Barret WAM^{TM} robot arm using a 200Hz vision setup which is currently being created at the Max-Planck Institute for Biological Cybernetics.

References to Directly Preceding Work:

A. J. Ijspeert, J. Nakanishi, and S. Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 1398–1403, Washington, DC, May 11-15 2002a. URL http://www-clmc.usc.edu/publications/I/ijspeert-ICRA2002.pdf .

A. J. Ijspeert, J. Nakanishi, and S. Schaal. Learning rhythmic movements by demonstration using nonlinear oscillators. In *Proceedings of the IEEE/RSJ 2002 International Conference on Intelligent RObots and Systems (IROS)*, pages 958–963, Lausanne, Sept.30-Oct.4 2002b. Piscataway, NJ: IEEE. URL http://www-clmc.usc.edu/publications/I/ijspeert-IROS2002.pdf.

A. J. Ijspeert, J. Nakanishi, and S. Schaal. Learning attractor landscapes for learning motor primitives. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 15, pages 1547–1554, Cambridge, MA, 2003. MIT Press. URL http://www-clmc.usc.edu/publications/I/ijspeert-NIPS2002.pdf.

J. Peters and S. Schaal. Learning operational space control. In W. Burgard, G. S. Sukhatme, and S. Schaal, editors, *Proceedings of Robotics: Science and Systems (R:SS)*. Cambridge, MA: MIT Press, 2006c. URL http://www-clmc.usc.edu/publications/P/peters-RSS2006.pdf.

J. Peters and S. Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2007b. URL http://www-clmc.usc.edu/publications/P/peters_ICML2007.pdf.

S. Schaal. The SL simulation and real-time control software package. Technical report, University of Southern California, 2004. URL http://www-clmc.usc.edu/publications/S/schaal-TRSL.pdf.

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die von mir am heutigen Tage eingereichte Diplomarbeit zum Thema

Reinforcement Learning for Motor Primitives

vollkommen selbstständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt wurden, sowie Zitate kenntlich gemacht habe.

Tübingen, den 21.08.2008

Jens Kober

Danksagung

Die vorliegende Arbeit entstand im Rahmen meiner Diplomarbeit am Max-Planck-Institut für biologische Kybernetik Tübingen, Abteilung Empirische Inferenz.

Danken möchte ich insbesonders Dr./USC Jan Peters für die Betreuung meiner Diplomarbeit am Institut, Prof. Dr.-Ing. Arnold Kistner für die Betreuung von Seiten der Universität Stuttgart, Prof. Dr. Bernhard Schölkopf für alle Unterstützung, meinen Labor-Kollegen Cand.-Ing. Katharina Mülling und Dipl.-Ing. Duy Nguyen-Tuong für die exzellente Zusammenarbeit, Dr./Univ. Utah Betty Mohler für die große Hilfe mit dem VICON^{TM} System, sowie den Kollegen am Instiut für die große Hilfsbereitschaft.

Außerdem danke ich meinen Eltern und meiner Familie für die Unterstützung und Begleitung von klein auf und meinem Freund Dipl.-Ing. Moritz Schöpfer für alle Unterstützung und für alles Verständnis.

Zusammenfassung

Bewegungsprimitive, die auf dynamischen Modellen beruhen [Ijspeert et al., 2002a], haben ermöglicht, dass Roboter komplexe Aufgaben von Tennisschlägen bis hin zum Robotergehen durch überwachtes Lernen beziehungsweise Regression lernen. Jedoch sind die meisten interessanten Lernprobleme hochdimensionale Reinforcement-Learning-Probleme (Bestärkendes Lernen), die oft mit den gängigen Methoden nicht gelöst werden können. Ergebnisse [Peters and Schaal, 2006b] zu Reinforcement-Learning mit unmittelbarer Bewertung werden für den episodischen Fall erweitert. Ein neuer Strategie-Lernalgorithmus wird hergeleitet, indem eine Form der Exploration verwendet wird, die besonders gut für Bewegungsprimitive geeignet ist. Der resultierende Algorithmus ist an Expectation-Maximization-Algorithmen angelehnt und lässt sich für das Lernen von komplexen Bewegungen einsetzen. Dieser Algorithmus wird mit mehreren bekannten Lernalgorithmen für parametrierte Strategien verglichen und es wird gezeigt, dass der neue Algorithmus diese sowie in Lerngeschwindigkeit als auch Endergebnis übertrifft. Der Algorithmus wird für das Lernen von Bewegungen verwendet und es wird gezeigt, dass damit die komplexe Bewegung des Spiels „Fangbecher“ auf einem realen Barrett WAM$^{\text{TM}}$ Roboterarm gelernt werden kann.

Die gelernten Steuerungs-Primitive können sehr empfindlich gegenüber Störeinflüssen bezüglich der Anfangsbedingungen oder der Trajektorie selbst sein. Um diese Störeinflüsse zu unterdrücken, ist die Rückführung der Wahrnehmung entscheidend. Jedoch gibt es bis jetzt nur wenige Ansätze, die Bewegungsprimitive um Wahrnehmungsrückführung von Variablen mit externem Fokus [Pongas et al., 2005] zu erweitern. Zudem beruhen diese Modifikationen auf Lösungen, die von Hand implementiert werden müssen. Menschen lernen, wie sie ihre Bewegungsprimitive bezüglich externer Variablen anpassen müssen. Es ist offensichtlich, dass eine solche Lösung in der Robotik benötigt wird. Daher wird eine Erweiterung der auf dynamischen Modellen basierenden Bewegungsprimitive entworfen, welche Wahrnehmungsrückführung integriert. Die daraus resultierenden Bewegungsprimitive, die von der Wahrnehmung beeinflusst werden, enthalten die ursprünglichen Bewegungsprimitive als Sonderfall und können so viele vorteilhafte Eigenschaften derer beibehalten. Es wird gezeigt, dass diese erweiterten Bewegungsprimitive komplexe Bewegungen wie das Fangbecherspiel sogar mit einer so großen Varianz in den Anfangsbedingungen meistern können, dass sie für einen geübten menschlichen Spieler eine Herausforderung wären. Um dies zu erreichen, werden die Bewegungsprimitive klassisch mit Imitationslernen ohne Wahrnehmungsrückführung initialisiert. Dann wird die Effizienz der Bewegungsprimitive mit der neuen Reinforcement-Learning-Methode verbessert.

Abstract

Motor primitives based on dynamical systems [Ijspeert et al., 2002a] have enabled robots to learn complex tasks ranging from tennis-swings to legged locomotion. However, most interesting motor learning problems are high-dimensional reinforcement learning problems often beyond the reach of current methods. We extend previous work [Peters and Schaal, 2006b] on policy learning from the immediate reward case to episodic reinforcement learning. We present a novel algorithm for policy learning by assuming a form of exploration that is particularly well-suited for dynamic motor primitives. The resulting algorithm is an EM-inspired algorithm applicable in complex motor learning tasks. We compare this algorithm to several well-known parametrized policy search methods and show that it outperforms them. We apply it in the context of motor learning and show that it can learn a complex Ball-in-a-Cup task using a real Barrett WAM$^{\text{TM}}$ robot arm.

The learned open loop policy trajectory can be very sensitive to perturbations of the initial conditions or the trajectory. Perceptual coupling is a natural choice to cancel these perturbations. However, to date there have been only few extensions which have incorporated perceptual coupling to variables of external focus [Pongas et al., 2005], and, furthermore, these modifications have relied upon handcrafted solutions. Humans learn how to couple their movement primitives with external variables. Clearly, such a solution is needed in robotics. We propose an augmented version of the motor primitives based on dynamical systems which incorporates perceptual coupling to an external variable. The resulting perceptually driven motor primitives include the previous primitives as a special case and can inherit some of their interesting properties. We show that these motor primitives can perform complex tasks such as Ball-in-a-Cup even with large variances in the initial conditions where a skilled human player would be challenged. For doing so, we initialize the motor primitives in the traditional way by imitation learning without perceptual coupling. Subsequently, we improve the motor primitives using our novel reinforcement learning method.

Contents

1 Introduction

Policy search, also known as policy learning, has become an accepted alternative of value function-based reinforcement learning [Bagnell et al., 2003, Strens and Moore, 2001, Kwee et al., 2001, Peshkin, 2001, El-Fakdi et al., 2006, Taylor et al., 2007]. In high-dimensional domains with continuous states and actions, such as robotics, this approach has previously proven successful as it allows the usage of domain-appropriate pre-structured policies, the straightforward integration of a teacher's presentation as well as fast online learning [Bagnell et al., 2003, Ng and Jordan, 2000, Peters and Schaal, 2006b, 2007b, Toussaint and Goerick, 2007, Hoffman et al., 2007, Guenter et al., 2007]. In this diploma thesis, we will extend the previous work in [Dayan and Hinton, 1997, Peters and Schaal, 2007b] from the immediate reward case to episodic reinforcement learning and show how it relates to policy gradient methods [Williams, 1992, Sutton et al., 2000, Lawrence et al., 2003, Tedrake et al., 2004, Peters and Schaal, 2006b]. Despite that many real-world motor learning tasks are essentially episodic [Wulf, 2007], episodic reinforcement learning [Sutton and Barto, 1998] is a largely undersubscribed topic. The resulting framework allows us to derive a new algorithm called Policy Learning by Weighting Exploration with the Returns (PoWER) which relies on a different stochastic policy than previous approaches. Instead of using additive state-independent, white Gaussian exploration in our motor command, we employ a state-dependent exploration strategy which is particularly well-suited for turning deterministic motor control policies into stochastic policies. We are especially interested in a particular kind of motor control policies also known as dynamic motor primitives [Ijspeert et al., 2003, Schaal et al., 2007]. In this approach, dynamical systems are being used in order to encode a policy, i.e., we have a special kind of parametrized policy which is well-suited for robotics problems.

The recent introduction of these motor primitives based on dynamical systems [Ijspeert et al., 2002a, 2003, Schaal et al., 2003, 2007] have allowed both imitation learning and reinforcement learning to acquire new behaviors fast and reliable. Resulting successes have shown that it is possible to rapidly learn motor primitives for complex behaviors such as tennis swings [Ijspeert et al., 2002a, 2003], T-ball batting [Peters and Schaal, 2006b], drumming [Pongas et al., 2005], biped locomotion [Schaal et al., 2003, Nakanishi et al., 2004b] and even in tasks with potential industrial application [Urbanek et al., 2004]. However, in their current form these motor primitives are generated in such a way that they are either only coupled to internal variables [Ijspeert et al., 2002a, 2003] or only include manually tuned phase-locking,

e.g., with an external beat [Pongas et al., 2005] or between the gait-generating primitive and the contact time of the feet [Schaal et al., 2003, Nakanishi et al., 2004b]. In many human motor control tasks, more complex perceptual coupling is needed in order to perform the task. Using handcrafted coupling based on human insight will in most cases no longer suffice.

In this diploma thesis, it is our goal to augment the Ijspeert-Nakanishi-Schaal approach [Ijspeert et al., 2002a, 2003] of using dynamical systems as motor primitives in such a way that it includes perceptual coupling with external variables. Similar to the biokinesiological literature on motor learning (see e.g., [Wulf, 2007]), we assume that there is an object of internal focus described by a state $\mathbf{x}$ and one of external focus $\mathbf{y}$. The coupling between both foci usually depends on the phase of the movement and, sometimes, the coupling only exists in short phases, e.g., in a catching movement, this could be at initiation of the movement (which is largely predictive) and during the last moment when the object is close to the hand (which is largely prospective or reactive and includes movement correction). Often, it is also important that the internal focus is in a different space than the external one. Fast movements, such as a Tennis-swing, always follow a similar pattern in joint-space of the arm while the external focus is usually on an object in Cartesian space or fovea-space. As a result, we have augmented the motor primitive framework in such a way that the coupling to the external, perceptual focus is phase-variant and both foci $\mathbf{y}$ and $\mathbf{x}$ can be in completely different spaces.

We show that the presented algorithm (PoWER) works well when employed in the context of learning dynamic motor primitives in four different settings, i.e., the two benchmark problems from [Peters and Schaal, 2006b], the Underactuated Swing-Up [Atkeson, 1994] and the complex task of Ball-in-a-Cup [Wikipedia, 2008a, Kawato et al., 1994]. Both the Underactuated Swing-Up as well as the Ball-in-a-Cup are achieved on a real Barrett WAM$^{\text{TM}}$ robot arm. Looking at these tasks from a human motor learning perspective, we have a human acting as teacher presenting an example for imitation learning and, subsequently, the policy will be improved by reinforcement learning. Since such tasks are inherently single-stroke movements, we focus on the special class of episodic reinforcement learning. In our experiments, we show how a presented movement which is recorded using kinesthetic teach-in or motion capture and, subsequently, how a Barrett WAM$^{\text{TM}}$ robot arm is learning the behavior by a combination of imitation and reinforcement learning.

Integrating perceptual coupling requires additional function approximation, and, as a result, the number of parameters of the motor primitives grows significantly. It becomes increasingly harder to manually tune these parameters to high performance and a learning approach for perceptual coupling is needed. The need for learning perceptual coupling in motor primitives has long been recognized in the motor primitive community [Schaal et al., 2007]. However, learning perceptual coupling to an external variable is not as straightforward. It requires many rollouts in

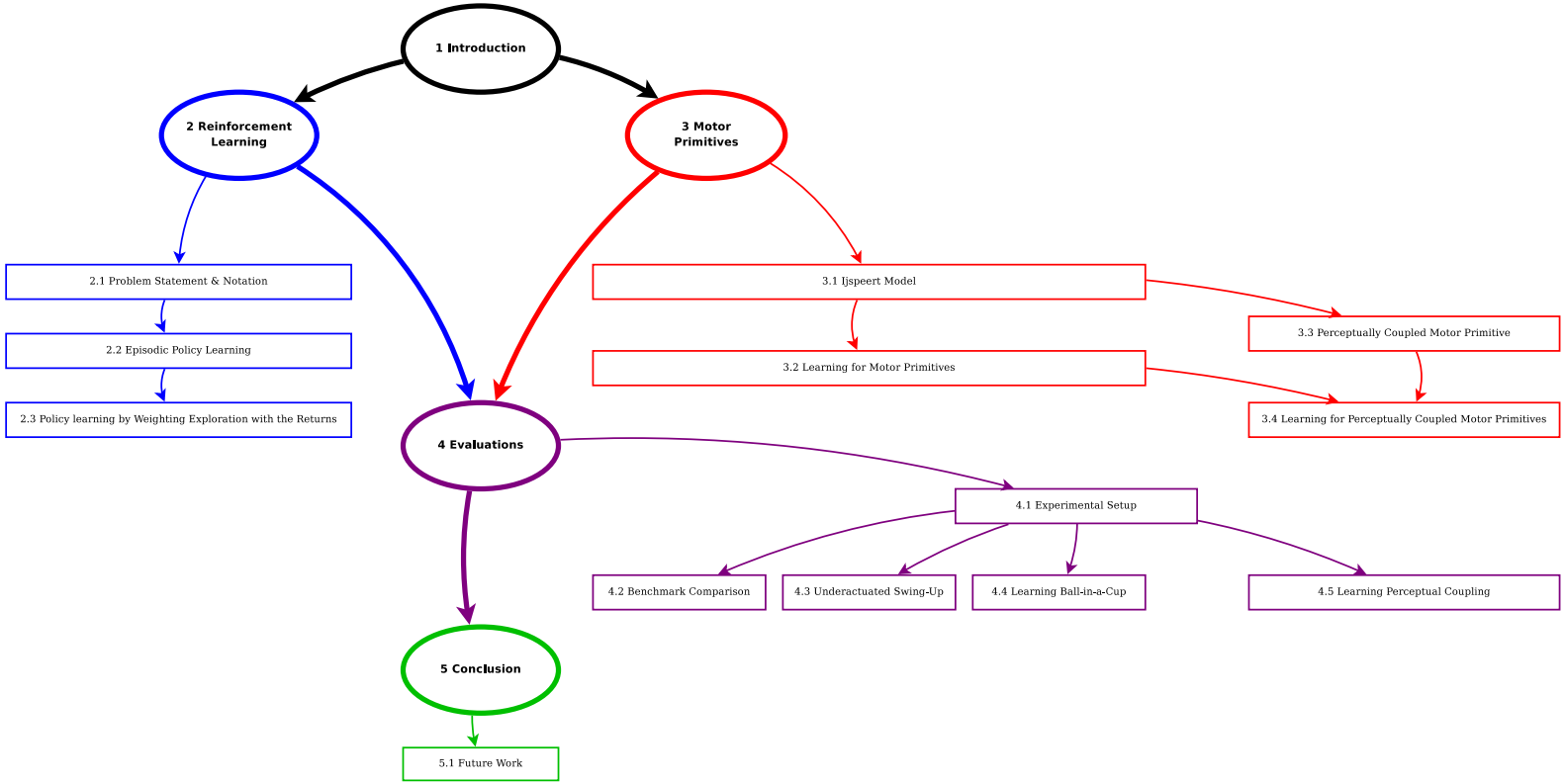


Figure 1.1: This figure shows the outline of this diploma-thesis. Sections 2 and 3 can be read independently. The evaluations in Section 4 are employing the reinforcement learning algorithm derived in Section 2 using Motor Primitives, described in Section 3, as parametrized policy.

order to properly determine the connections from external to internal focus. It is straightforward to grasp a general movement by imitation and a human can produce a Ball-in-a-Cup movement or a Tennis-swing after a single or few observed trials of a teacher but he will never have a robust coupling to the ball. Furthermore, small differences between the kinematics of teacher and student amplify in the perceptual coupling. This part is the reason why perceptually driven motor primitives can be initialized by imitation learning but will usually require self-improvement by reinforcement learning. This approach is analogous to the case of a human learning tennis: a teacher can present a forehand to the human student but a lot of self-practice is needed for a proper tennis game. We again use PoWER to learn the parameters for the perceptual coupling.

The outline of this diploma-thesis is shown in Figure 1.1. In Section 2 we introduce a framework for episodic reinforcement learning algorithms. Using this approach we derive the policy gradient theorem [Sutton et al., 2000], a generalization of the reward-weighted regression [Peters and Schaal, 2007b] as well as the novel Policy learning by Weighting Exploration with the Returns (PoWER) algorithm. In Section 3, we discuss the [Ijspeert et al., 2002a, 2003, Schaal et al., 2007] motor primitives based on dynamical systems as well as our augmentation for perceptual coupling. We also analyze a suitable imitation learning algorithm and how reinforcement learning can be applied. The evaluations in Section 4 are employing the reinforcement learning algorithm derived in Section 2 using motor primitives, described in Section 3, as parametrized policy. For the reinforcement learning comparison, we study two benchmark problems with the introduced algorithms and Ball-in-a-Cup with our novel method. Perceptual coupling is evaluated for the Ball-in-a-Cup example. The evaluations are done in simulation and on a real Barrett WAM$^{\text{TM}}$ robot arm.

Ball-in-a-Cup or Kendama (see Section 4.4) has previously been studied in robotics. Takenaka [1984] used a three degree of freedom robot that is moving on a plane to perform Ball-in-a-Cup. The robot is controlled by a trajectory tracking controller based on a mathematical model. The trajectory of the end effector is taken from human patterns. Cameras are used for nonlinear compensations. Sakaguchi and Miyazaki [1994], Sato et al. [1993] take a distinctively different approach without imitation learning. They use a two degree of freedom robot (also moving on a plane) that receives feedback from force sensors (which corresponds to playing Ball-in-a-Cup with eyes closed). A model of the pendulum is given. The motion planning is divided in sub tasks. First, the robot performs some movements in order to estimate the parameters of the model (mass and string length). Subsequently the parameters for the trajectory are calculated based on the model. Finally, the trajectory is optimized based on this feedback. Miyamoto et al. [1996] used a seven degree of freedom anthropomorphic arm. They record human motions which is used to train a neural network to reconstruct via-points. The imitation is designed to match the Cartesian coordinates exactly and the joints as closely as possible. The motion is

performed open-loop and visual feedback is used to improve the trajectory using a Newton-like algorithm. Shone et al. [2000] construct a simple one degree of freedom robot particularly for Ball-in-a-Cup. The trajectory is found using a gradient based path planner in simulation. Ball-in-a-Cup is modeled as pendulum switching to a free point mass. The found trajectory is executed on the real system, which only has the motor encoders as feedback. There exists also further related work: Fujii et al. [1993] model Kendama using a number of approximations (string as line segments, no string tension), Arisumi et al. [2005] use a casting manipulator to throw the ball on the spike. Ball-in-a-Cup has also been studied in space by NASA [Sumners, 1997] for educational purposes and is used to study the relation of sleep and learning by Milner et al. [2006], Fogel and Smith [2006].

2 Reinforcement Learning

Our goal is to find reinforcement learning techniques that can be applied to a special kind of pre-structured parametrized policies called *motor primitives* [Ijspeert et al., 2003, Schaal et al., 2007] (also described in Section 3.1), in the context of learning high-dimensional motor control tasks. In order to do so, we first discuss our problem in the general context of reinforcement learning and introduce the required notation in Section 2.1. Using a generalization of the approach in [Dayan and Hinton, 1997, Peters and Schaal, 2007b], we derive a new expectation-maximization (EM) inspired algorithm [Dempster et al., 1977] called Policy Learning by Weighting Exploration with the Returns (PoWER) in Section 2.3 and show how the general framework is related to policy gradients methods in 2.2. Attias [2003] extends the [Dayan and Hinton, 1997] algorithm to episodic reinforcement learning for discrete states; we use continuous states. Subsequently, we discuss how we can turn the parametrized motor primitives [Ijspeert et al., 2003, Schaal et al., 2007] into stochastic policies. For doing so, we do not follow previous work which perturbs motor commands based on additive, state-independent exploration [Williams, 1992, Guenter et al., 2007, Peters and Schaal, 2006b] but instead use a state dependent exploration which allows the derivation of algorithms with faster convergence to optimal solutions.

2.1 Problem Statement & Notation

In this diploma thesis, we treat motor primitive learning problems in the framework of reinforcement learning with a strong focus on the episodic case [Sutton and Barto, 1998]. We assume that at time t there is an actor in a state $\mathbf{s}_t$ and chooses an appropriate action $\mathbf{a}_t$ according to a stochastic policy $\pi(\mathbf{a}_t|\mathbf{s}_t, t)$. Such a policy is a probability distribution over actions given the current state. The stochastic formulation allows a natural incorporation of exploration and, in the case of hidden state variables, the optimal time-invariant policy has been shown to be stochastic [Sutton et al., 2000]. Upon the completion of the action, the actor transfers to a state $\mathbf{s}_{t+1}$ and receives a reward r_t. As we are interested in learning complex motor tasks consisting of a single stroke [Schaal et al., 2007], we focus on finite horizons of length T with episodic restarts [Sutton and Barto, 1998] and learn the optimal parametrized, stochastic policy for such reinforcement learning problems. We assume an explorative version of the dynamic motor primitives [Ijspeert et al.,

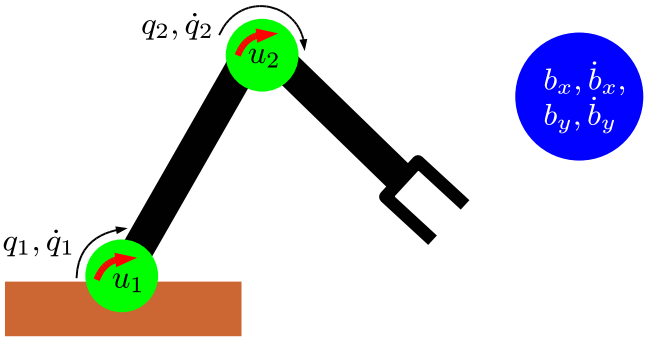


Figure 2.1: This illustration shows a planar robot with two degrees of freedom. In this example the states $\mathbf{s} = [q_1, \dot{q}_1, q_2, \dot{q}_2, b_x, \dot{b}_x, b_y, \dot{b}_y]$ include the joint angles q_n and the angular joint velocities $\dot{q}_n$ as well as the Cartesian positions b_n and velocities $\dot{b}_n$ of an external object. The states describe the momentary state of the system. Actions $\mathbf{a} = [u_1, u_2]$, which correspond to motor torques, are used to influence the system.

2003, Schaal et al., 2007] (see Section 3.1) as parametrized policy π with parameters $\boldsymbol{\theta} \in \mathbb{R}^n$. However, in this section, we will keep most derivations sufficiently general that they would transfer to various other parametrized policies. The general goal in reinforcement learning is to optimize the *expected return* of the policy π with parameters $\boldsymbol{\theta}$ defined by

$$J(\boldsymbol{\theta}) = \int_{\mathbb{T}} p(\boldsymbol{\tau}) R(\boldsymbol{\tau}) d\boldsymbol{\tau}, \tag{2.1}$$

where $\mathbb{T}$ is the set of all possible paths, rollout $\boldsymbol{\tau} = [\mathbf{s}_{1:T+1}, \mathbf{a}_{1:T}]$ (also called episode or trial) denotes a path of states $\mathbf{s}_{1:T+1} = [\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_{T+1}]$ and actions $\mathbf{a}_{1:T} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_T]$. The probability of rollout $\boldsymbol{\tau}$ is denoted by $p(\boldsymbol{\tau})$ while $R(\boldsymbol{\tau})$ refers to its return. Using the standard assumptions of Markovness and additive accumulated rewards, we can write

$$p(\boldsymbol{\tau}) = p(\mathbf{s}_1) \prod_{t=1}^{T} p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) \pi(\mathbf{a}_t|\mathbf{s}_t, t), \tag{2.2}$$

$$R(\boldsymbol{\tau}) = T^{-1} \sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}, t), \tag{2.3}$$

where $p(\mathbf{s}_1)$ denotes the initial state distribution, $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ the next state distribution conditioned on last state and action, and $r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}, t)$ denotes the immediate reward. See Figures 2.1 and 2.2 for illustrations.

While episodic reinforcement learning (RL) problems with finite horizons are common in motor control, few methods exist in the RL literature, e.g., Episodic REINFORCE [Williams, 1992], the Episodic Natural Actor Critic eNAC [Peters and Schaal, 2006b]

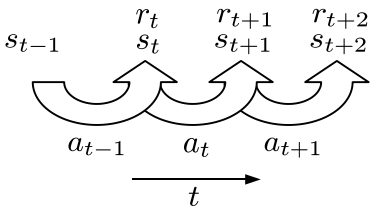


Figure 2.2: Starting at state s_t action a_t is applied. This results in state s_{t+1} in the next time step. States are summarized as $\mathbf{s}_{1:T+1} = [s_1, s_2, \ldots, s_{T+1}]$ and actions as $\mathbf{a}_{1:T} = [a_1, a_2, \ldots, a_T]$. The combined information from states and actions is called rollout $\boldsymbol{\tau} = [\mathbf{s}_{1:T+1}, \mathbf{a}_{1:T}]$. A reward r_t, based on the values of s_t, a_t, s_{t+1} and t, is given in each time step. The rewards are summarized as $\mathbf{r}_{1:T} = [r_1, r_2, \ldots, r_T]$. The policy $\pi(a_t|s_t, t)$, which is a probability distribution dependent on the current state and time step, describes the next action. We use a parametrized policy with parameters $\boldsymbol{\theta} \in \mathbb{R}^n$.

and model-based methods using differential-dynamic programming [Atkeson, 1994]. Nevertheless, in the analytically tractable cases, it has been studied deeply in the optimal control community where it is well-known that for a finite horizon problem, the optimal solution is non-stationary [Kirk, 1970] and, in general, cannot be represented by a time-independent policy. The motor primitives based on dynamical systems [Ijspeert et al., 2003, Schaal et al., 2007] are a particular type of time-variant policy representation as they have an internal phase which corresponds to a clock with additional flexibility (e.g., for incorporating coupling effects, perceptual influences, etc.), thus, they can represent optimal solutions for finite horizons. We embed this internal clock or movement phase into our state and, thus, from optimal control perspective have ensured that the optimal solution can be represented.

2.2 Episodic Policy Learning

In this section, we discuss episodic reinforcement learning in policy space which we will refer to as Episodic Policy Learning. For doing so, we first discuss the lower bound on the expected return suggested in [Dayan and Hinton, 1997] for guaranteeing that policy update steps are improvements. In [Dayan and Hinton, 1997, Peters and Schaal, 2007b] only the immediate reward case is being discussed, we extend their framework to episodic reinforcement learning and, subsequently, derive a general update rule which yields the policy gradient theorem [Sutton et al., 2000], a generalization of the reward-weighted regression [Peters and Schaal, 2007b] as well as the novel Policy learning by Weighting Exploration with the Returns (PoWER) algorithm.

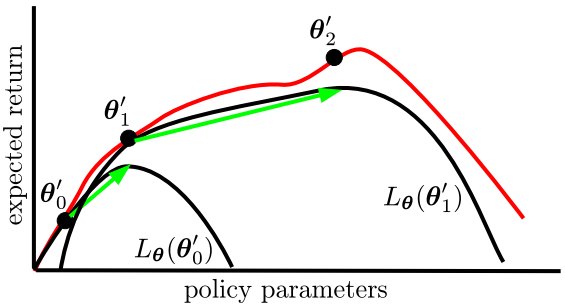


Figure 2.3: This figure shows an illustration of the maximization of the lower bounds (black) and the resulting parameter updates (green).

2.2.1 Bounds on Policy Improvements

Unlike in reinforcement learning, most other machine learning branches have focused on optimizing lower bounds, e.g., resulting in expectation-maximization (EM) algorithms [McLachan and Krishnan, 1997]. The reasons for this preference apply in policy learning: if the lower bound also becomes an equality for the sampling policy, we can guarantee that the policy will be improved by optimizing the lower bound. Surprisingly, results from supervised learning can be transferred with ease. For doing so, we follow the scenario suggested in [Dayan and Hinton, 1997], i.e., generate rollouts $\boldsymbol{\tau}$ using the current policy with parameters $\boldsymbol{\theta}$ which we weight with the returns $R(\boldsymbol{\tau})$ and subsequently match it with a new policy parametrized by $\boldsymbol{\theta}'$. This matching of the success-weighted path distribution is equivalent to minimizing the Kullback-Leibler divergence $D(p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}) \| p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) r(\boldsymbol{\tau}))$ between the new path distribution $p_{\boldsymbol{\theta}'}(\boldsymbol{\tau})$ and the reward-weighted previous one $p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) r(\boldsymbol{\tau})$. As shown in [Dayan and Hinton, 1997, Peters and Schaal, 2007b], this results in a lower bound on the expected return using Jensen's inequality and the concavity of the logarithm, i.e.,

$$\log J(\boldsymbol{\theta}') = \log \int_{\mathbb{T}} \frac{p_{\boldsymbol{\theta}}(\boldsymbol{\tau})}{p_{\boldsymbol{\theta}}(\boldsymbol{\tau})} p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}) R(\boldsymbol{\tau}) d\boldsymbol{\tau}, \tag{2.4}$$

$$\geq \int_{\mathbb{T}} p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) R(\boldsymbol{\tau}) \log \frac{p_{\boldsymbol{\theta}'}(\boldsymbol{\tau})}{p_{\boldsymbol{\theta}}(\boldsymbol{\tau})} d\boldsymbol{\tau} + \text{const}, \tag{2.5}$$

$$\propto -D(p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}) \| p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) R(\boldsymbol{\tau})) = L_{\boldsymbol{\theta}}(\boldsymbol{\theta}'), \tag{2.6}$$

where

$$D(p(\boldsymbol{\tau}) \| q(\boldsymbol{\tau})) = \int p(\boldsymbol{\tau}) \log \left(\frac{p(\boldsymbol{\tau})}{q(\boldsymbol{\tau})} \right) d\boldsymbol{\tau} \tag{2.7}$$

is the Kullback-Leibler divergence which is considered a natural distance measure between probability distributions, and the constant is needed for tightness of the

bound. Note that $p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) R(\boldsymbol{\tau})$ is an improper probability distribution as pointed out in [Dayan and Hinton, 1997]. Maximizing the lower bound on the expected return $L_{\boldsymbol{\theta}}(\boldsymbol{\theta}') = -D(p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}) \| p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) R(\boldsymbol{\tau}))$ is the essential step when improving a policy and we will show the relation to different previous policy learning methods in Section 2.2.2. See Figure 2.3 for an illustration.

2.2.2 Resulting Policy Updates

In the following part, we will discuss three different policy updates which directly result from Section 2.2.1. First, we show that policy gradients [Williams, 1992, Sutton et al., 2000, Lawrence et al., 2003, Tedrake et al., 2004, Peters and Schaal, 2006b] can be derived from the lower bound $L_{\boldsymbol{\theta}}(\boldsymbol{\theta}')$. Subsequently, we show that natural policy gradients [Peters and Schaal, 2006b] can be seen as an additional constraint regularizing the change in the path distribution resulting from a policy update when improving the policy incrementally. Finally, we will show how expectation-maximization (EM) algorithms for policy learning can be generated.

Policy Gradients. When differentiating the function $L_{\boldsymbol{\theta}}(\boldsymbol{\theta}')$ that defines the lower bound on the expected return, we directly obtain

$$\partial_{\boldsymbol{\theta}'} L_{\boldsymbol{\theta}}(\boldsymbol{\theta}') = \int_{\mathbb{T}} p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) R(\boldsymbol{\tau}) \partial_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}) d\boldsymbol{\tau}, \tag{2.8}$$

where $\mathbb{T}$ is the set of all possible paths and

$$\partial_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}) = \sum_{t=1}^{T} \partial_{\boldsymbol{\theta}} \log \pi(\mathbf{a}_t | \mathbf{s}_t, t) \tag{2.9}$$

denotes the log-derivative of the path distribution. As this log-derivative only depends on the policy, we can estimate a gradient from roll-outs without having a model by simply replacing the expectation by a sum; when $\boldsymbol{\theta}'$ is close to $\boldsymbol{\theta}$, we have the policy gradient estimator which is widely known as Episodic REINFORCE [Williams, 1992], i.e., we have $\lim_{\boldsymbol{\theta}' \to \boldsymbol{\theta}} \partial_{\boldsymbol{\theta}'} L_{\boldsymbol{\theta}}(\boldsymbol{\theta}') = \partial_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$. Obviously, a reward which precedes an action in an rollout, can neither be caused by the action nor cause an action in the same rollout. Thus, when inserting Equations (2.2,2.3) into Equation (2.8), all cross-products between r_t and $\partial_{\boldsymbol{\theta}} \log \pi(\mathbf{a}_{t+\delta t} | \mathbf{s}_{t+\delta t}, t + \delta t)$ for $\delta t > 0$ become zero in expectation [Peters and Schaal, 2006b]. Therefore, we can omit these terms and rewrite the estimator as

$$\partial_{\boldsymbol{\theta}'} L_{\boldsymbol{\theta}}(\boldsymbol{\theta}') = E\left\{ \sum_{t=1}^{T} \partial_{\boldsymbol{\theta}} \log \pi(\mathbf{a}_t | \mathbf{s}_t, t) Q^{\pi}(\mathbf{s}, \mathbf{a}, t) \right\}, \tag{2.10}$$

Algorithm 2.1 Finite Difference Gradients (FDG)

Input: initial policy parameters $\boldsymbol{\theta}_0$

repeat

Generate policy variations:

$$\boldsymbol{\Delta\theta}^h \sim \mathcal{U}_{[-\boldsymbol{\Delta\theta}_{\min}, \boldsymbol{\Delta\theta}_{\max}]}$$

for $h = \{1, \ldots, H\}$ rollouts.

Sample:
Perform $h = \{1, \ldots, H\}$ rollouts using

$$\mathbf{a} = \left(\boldsymbol{\theta} + \boldsymbol{\Delta\theta}^h\right)^{\mathrm{T}} \boldsymbol{\phi}\left(\mathbf{s}, t\right)$$

as policy and collect all
$\left(t, \mathbf{s}_t^h, \mathbf{a}_t^h, \mathbf{s}_{t+1}^h, \boldsymbol{\varepsilon}_t^h, r_{t+1}^h\right)$ for $t = \{1, 2, \ldots, T+1\}$.

Compute:
Return

$$R^h\left(\boldsymbol{\theta} + \boldsymbol{\Delta\theta}^h\right) = \sum_{t=1}^{T+1} r_t^h$$

from rollouts.

Compute Gradient:

$$\left[\begin{array}{cc} \mathbf{g}_{\mathrm{FD}}^{\mathrm{T}} & , R_{\mathrm{ref}} \end{array}\right]^{\mathrm{T}} = \left(\boldsymbol{\Delta\Theta}^{\mathrm{T}} \boldsymbol{\Delta\Theta}\right)^{-1} \boldsymbol{\Delta\Theta}^{\mathrm{T}} \mathbf{R}$$

with $\boldsymbol{\Delta\Theta} = \left[\begin{array}{ccc} \boldsymbol{\Delta\theta}^1, & \ldots, & \boldsymbol{\Delta\theta}^H \\ 1, & \ldots, & 1 \end{array}\right]^{\mathrm{T}}$
and $\mathbf{R} = \left[\begin{array}{ccc} R^1, & \ldots, & R^H \end{array}\right]^{\mathrm{T}}$.

Update:
policy using

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha \mathbf{g}_{\mathrm{FD}}.$$

until Convergence $\boldsymbol{\theta}_{k+1} \approx \boldsymbol{\theta}_k$.

Algorithm 2.2 'Vanilla' Policy Gradients (VPG)

Input: initial policy parameters $\boldsymbol{\theta}_0$

repeat

Sample:
Perform $h = \{1, \ldots, H\}$ rollouts using

$$\mathbf{a} = \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\phi}\left(\mathbf{s}, t\right) + \boldsymbol{\varepsilon}_t$$

with $[\varepsilon_t^n] \sim \mathcal{N}\left(0, \left(\sigma^{h,n}\right)^2\right)$ as stochastic policy and collect all $\left(t, \mathbf{s}_t^h, \mathbf{a}_t^h, \mathbf{s}_{t+1}^h, \boldsymbol{\varepsilon}_t^h, r_{t+1}^h\right)$ for $t = \{1, 2, \ldots, T+1\}$.

Compute:
Return

$$R^h = \sum_{t=1}^{T+1} r_t^h,$$

eligibility

$$\psi^{h,n} = \frac{\partial \log p\left(\boldsymbol{\tau}^h\right)}{\partial \theta^n} = \sum_{t=1}^{T} \frac{\partial \log \pi\left(a_t^h | s_t^h, t\right)}{\partial \theta^n} = \sum_{t=1}^{T} \frac{1}{\left(\sigma_h^n\right)^2} \varepsilon_t^{h,n} \phi^n\left(s_t^{h,n}, t\right)$$

and baseline

$$b^n = \frac{\sum_{h=1}^{H} \left(\psi^{h,n}\right)^2 R^h}{\sum_{h=1}^{H} \left(\psi^{h,n}\right)^2}$$

for each parameter $n = \{1, \ldots, N\}$ from rollouts.

Compute Gradient:

$$g_{\mathrm{VP}}^n = E\left\{\frac{\partial \log p\left(\boldsymbol{\tau}^h\right)}{\partial \theta^n}\left(r\left(\boldsymbol{\tau}^h\right) - b^n\right)\right\} = \frac{1}{H}\sum_{h=1}^{H} \psi^{h,n}\left(R^h - b^n\right).$$

Update:
policy using

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha \mathbf{g}_{\mathrm{VP}}.$$

until Convergence $\boldsymbol{\theta}_{k+1} \approx \boldsymbol{\theta}_k$.

where

$$Q^{\pi}(\mathbf{s}, \mathbf{a}, t) = E\left\{\sum_{\tilde{t}=t}^{T} r\left(\mathbf{s}_{\tilde{t}}, \mathbf{a}_{\tilde{t}}, \mathbf{s}_{\tilde{t}+1}, \tilde{t}\right) | \mathbf{s}_t = \mathbf{s}, \mathbf{a}_t = \mathbf{a}\right\} \tag{2.11}$$

is called the state-action value function [Sutton and Barto, 1998]. Equation (2.10) is equivalent to the policy gradient theorem [Sutton et al., 2000] for $\boldsymbol{\theta}' \rightarrow \boldsymbol{\theta}$ in the infinite horizon case where the dependence on time t can be dropped. [Binder et al., 1997] also show from a different perspective the equivalence of Generalized EM and Gradient Descent. See Algorithms 2.1 and 2.2 for exemplary implementations.

The derivation results in the Natural Actor Critic as discussed in [Bagnell and Schneider, 2003, Peters et al., 2003a, 2005, Peters and Schaal, 2006b, 2008b] when adding an additional punishment to prevent large steps away from the observed path distribution. This can be achieved by restricting the amount of change in the path distribution and, subsequently, determining the steepest descent for a fixed step away from the observed trajectories. Change in probability distributions is naturally measured using the Kullback-Leibler divergence [Peters and Schaal, 2008b], thus, after adding the additional constraint of $D\left(p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}) \| p_{\boldsymbol{\theta}}(\boldsymbol{\tau})\right) = \delta$, we can derive the natural policy gradient by simply approximating

$$D\left(p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}) \| p_{\boldsymbol{\theta}}(\boldsymbol{\tau})\right) \approx 0.5\left(\boldsymbol{\theta}' - \boldsymbol{\theta}\right)^{\mathrm{T}} \mathbf{F}(\boldsymbol{\theta})\left(\boldsymbol{\theta}' - \boldsymbol{\theta}\right) \tag{2.12}$$

with its second-order expansion where $\mathbf{F}(\boldsymbol{\theta})$ denotes the Fisher information matrix [Bagnell and Schneider, 2003, Peters et al., 2003a]. See Algorithm 2.3 for an exemplary implementation.

Policy Search via Expectation Maximization. One major drawback of gradient-based approaches is the learning rate, an open parameter which can be hard to tune in control problems but is essential for good performance. Expectation-Maximization algorithms are well-known to avoid this problem in supervised learning while even yielding second-order convergence [McLachan and Krishnan, 1997]. Previously, similar ideas have been explored in immediate reinforcement learning [Dayan and Hinton, 1997, Peters and Schaal, 2007c, 2008a, 2006c, 2007b]. In general, an EM-algorithm would choose the next policy parameters $\boldsymbol{\theta}_{n+1}$ such that $\boldsymbol{\theta}_{n+1} = \mathrm{argmax}_{\boldsymbol{\theta}'} L_{\boldsymbol{\theta}}(\boldsymbol{\theta}')$. In the case where $\pi(\mathbf{a}_t|\mathbf{s}_t, t)$ belongs to the exponential family, the next policy can be determined analytically by setting Equation (2.10) to zero, i.e.,

$$E\left\{\sum_{t=1}^{T} \partial_{\boldsymbol{\theta}'} \log \pi\left(\mathbf{a}_t|\mathbf{s}_t, t\right) Q^{\pi}\left(\mathbf{s}, \mathbf{a}, t\right)\right\} = 0, \tag{2.13}$$

and solving for $\boldsymbol{\theta}'$. Depending on the choice of a stochastic policy, we will obtain different solutions and different learning algorithms. It allows the extension of the reward-weighted regression to larger horizons (see Algorithm 2.4 for an exemplary implementation) as well as the introduction of the Policy learning by Weighting Exploration with the Returns (PoWER) algorithm.

Algorithm 2.3 episodic Natural Actor Critic (eNAC)

Input: initial policy parameters $\boldsymbol{\theta}_0$

repeat

Sample:
Perform $h = \{1, \ldots, H\}$ rollouts using

$$\mathbf{a} = \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\phi}\left(\mathbf{s}, t\right) + \boldsymbol{\varepsilon}_t$$

with $[\varepsilon_t^n] \sim \mathcal{N}\left(0, \left(\sigma^{h,n}\right)^2\right)$ as stochastic policy and collect all $\left(t, \mathbf{s}_t^h, \mathbf{a}_t^h, \mathbf{s}_{t+1}^h, \boldsymbol{\varepsilon}_t^h, r_{t+1}^h\right)$ for $t = \{1, 2, \ldots, T+1\}$.

Compute:
Return

$$R^h = \sum_{t=1}^{T+1} r_t^h$$

and eligibility

$$\psi^{h,n} = \sum_{t=1}^{T} \frac{1}{\left(\sigma_h^n\right)^2} \varepsilon_t^{h,n} \phi^n \left(s_t^{h,n}, t\right)$$

for each parameter $n = \{1, \ldots, N\}$ from rollouts.

Compute Gradient:

$$\left[\begin{array}{c} \mathbf{g}_{\mathrm{eNAC}}^{\mathrm{T}}, \quad R_{\mathrm{ref}} \end{array}\right]^{\mathrm{T}} = \left(\boldsymbol{\Psi}^{\mathrm{T}} \boldsymbol{\Psi}\right)^{-1} \boldsymbol{\Psi}^{\mathrm{T}} \mathbf{R}$$

with $\boldsymbol{\Psi} = \left[\begin{array}{ccc} \boldsymbol{\psi}^1, & \ldots, & \boldsymbol{\psi}^H \\ 1, & \ldots, & 1 \end{array}\right]^{\mathrm{T}}$
and $\mathbf{R} = \left[\begin{array}{ccc} R^1, & \ldots, & R^H \end{array}\right]^{\mathrm{T}}$.

Update:
policy using

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha \mathbf{g}_{\mathrm{eNAC}}.$$

until Convergence $\boldsymbol{\theta}_{k+1} \approx \boldsymbol{\theta}_k$.

Algorithm 2.4 episodic Reward Weighted Regression (RWR)

Input: initial policy parameters $\boldsymbol{\theta}_0$

repeat

Sample:
Perform $h = \{1, \ldots, H\}$ rollouts using

$$\mathbf{a} = \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\phi}\left(\mathbf{s}, t\right) + \boldsymbol{\varepsilon}_t$$

with $[\varepsilon_t^n] \sim \mathcal{N}\left(0, \left(\sigma^{h,n}\right)^2\right)$ as stochastic policy and collect all $\left(t, \mathbf{s}_t^h, \mathbf{a}_t^h, \mathbf{s}_{t+1}^h, \boldsymbol{\varepsilon}_t^h, r_{t+1}^h\right)$ for $t = \{1, 2, \ldots, T+1\}$.

Compute:
Return

$$R^h = \sum_{t=1}^{T+1} r_t^h$$

from rollouts.

Update:
policy using

$$\boldsymbol{\theta}_{k+1}^n = \left(\left(\boldsymbol{\Phi}^n\right)^{\mathrm{T}} \mathbf{R} \boldsymbol{\Phi}^n\right)^{-1} \left(\boldsymbol{\Phi}^n\right)^{\mathrm{T}} \mathbf{R} \mathbf{A}^n$$

with basis functions $\boldsymbol{\Phi}^n = \left[\phi_1^{1,n}, \ldots, \phi_T^{1,n}, \phi_1^{2,n}, \ldots, \phi_T^{2,n}, \ldots, \phi_1^{H,n}, \ldots, \phi_T^{H,n}\right]^{\mathrm{T}}$,
actions $\mathbf{A}^n = \left[a_1^{1,n}, \ldots, a_T^{1,n}, a_1^{2,n}, \ldots, a_T^{2,n}, \ldots, a_1^{H,n}, \ldots, a_T^{H,n}\right]^{\mathrm{T}}$
and returns $\mathbf{R} = \mathrm{diag}\left(R^1, \ldots, R^1, R^2, \ldots, R^2, \ldots, R^H, \ldots, R^H\right)$.

until Convergence $\boldsymbol{\theta}_{k+1} \approx \boldsymbol{\theta}_k$.

2.3 Policy Learning by Weighting Exploration with the Returns

In most learning control problems, we attempt to have a deterministic mean policy $\bar{\mathbf{a}} = \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{s}, \mathbf{t})$ with parameters $\boldsymbol{\theta}$ and basis functions $\boldsymbol{\phi}$, e.g., in linear-quadratic regulation where we have gains as $\boldsymbol{\theta}$ and states as $\boldsymbol{\phi}(\mathbf{s}, \mathbf{t})$. In Section 3.1, we will introduce the basis functions of the motor primitives. When learning motor primitives, we turn this deterministic mean policy $\bar{\mathbf{a}} = \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{s}, t)$ into a stochastic policy using additive exploration $\boldsymbol{\varepsilon}(\mathbf{s}, t)$ in order to make model-free reinforcement learning possible, i.e., we always intend to have a policy $\pi(\mathbf{a}_t|\mathbf{s}_t, t)$ which can be brought into the form

$$\mathbf{a} = \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{s}, t) + \boldsymbol{\epsilon}(\boldsymbol{\phi}(\mathbf{s}, t)). \tag{2.14}$$

Previous work in this context [Williams, 1992, Guenter et al., 2007, Peters and Schaal, 2006b, 2007b] has focused on state-independent, white Gaussian exploration, i.e., $\boldsymbol{\epsilon}(\boldsymbol{\phi}(\mathbf{s}, t)) \sim \mathcal{N}(0, \Sigma)$. It is straightforward to obtain the Reward-Weighted Regression for episodic RL by solving Equation (2.13) for $\boldsymbol{\theta}'$ which naturally yields a weighted regression method with the state-action values $Q^{\pi}(\mathbf{s}, \mathbf{a}, t)$ as weights. This form of exploration has resulted into various applications in robotics such as tennis swings [Ijspeert et al., 2002a, 2003], T-Ball batting [Peters and Schaal, 2006b], Peg-In-Hole [Gullapalli et al., 1994], humanoid robot locomotion [Schaal et al., 2003, Nakanishi et al., 2004b], constrained reaching movements [Guenter et al., 2007] and operational space control [Peters and Schaal, 2007c], see [Guenter et al., 2007, Peters and Schaal, 2006b, 2007b] for both reviews and their own applications.

However, such unstructured exploration at every step has a multitude of disadvantages: it causes a large variance which grows with the number of time-steps [Peters and Schaal, 2006b], it perturbs actions too frequently 'washing' out their effects and can damage the system executing the trajectory. As a result, all methods relying on this state-independent exploration have proven too fragile for learning the Ball-in-a-Cup task on a real robot system. Alternatively, one could generate a form of structured, state-dependent exploration

$$\boldsymbol{\epsilon}(\boldsymbol{\phi}(\mathbf{s}, t)) = \boldsymbol{\varepsilon}_t^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{s}, t) \tag{2.15}$$

with $[\boldsymbol{\varepsilon}_t]_{ij} \sim \mathcal{N}(0, \sigma_{ij}^2)$, where σ_{ij}^2 are meta-parameters of the exploration that can also be optimized. This argument results into the policy

$$\mathbf{a} \sim \pi(\mathbf{a}_t|\mathbf{s}_t, t) = \mathcal{N}\left(\mathbf{a}|\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{s}, t), \hat{\boldsymbol{\Sigma}}(\mathbf{s}, t)\right). \tag{2.16}$$

Inserting the resulting policy into Equation (2.13), we obtain the optimality condition update in the sense of Equation (2.13) and can derive the update rule

$$\boldsymbol{\theta}' = \boldsymbol{\theta} + \frac{E\left\{\sum_{t=1}^{T} \boldsymbol{\varepsilon}_t Q^{\pi}(\mathbf{s}, \mathbf{a}, t)\right\}}{E\left\{\sum_{t=1}^{T} Q^{\pi}(\mathbf{s}, \mathbf{a}, t)\right\}}. \tag{2.17}$$

Algorithm 2.5 EM Policy learning by Weighting Exploration with the Returns (PoWER)

Input: initial policy parameters $\boldsymbol{\theta}_0$

repeat

Sample:
Perform rollout(s) using

$$\mathbf{a} = \left(\boldsymbol{\theta} + \boldsymbol{\varepsilon}_t\right)^{\mathrm{T}} \boldsymbol{\phi}\left(\mathbf{s}, t\right)$$

with $\left[\boldsymbol{\varepsilon}_t\right]_{ij} \sim \mathcal{N}\left(0, \sigma_{ij}^2\right)$ as stochastic policy and collect all $\left(t, \mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}, \boldsymbol{\varepsilon}_t, r_{t+1}\right)$ for $t = \{1, 2, \ldots, T+1\}$.

Estimate:
Use unbiased estimate

$$\hat{Q}^{\pi}\left(\mathbf{s}, \mathbf{a}, t\right) = \sum_{\tilde{t}=t}^{T} r\left(\mathbf{s}_{\tilde{t}}, \mathbf{a}_{\tilde{t}}, \mathbf{s}_{\tilde{t}+1}, \tilde{t}\right).$$

Reweight:
Compute importance weights and reweight rollouts, discard low-importance rollouts.

Update:
policy using

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \frac{\left\langle \sum_{t=1}^{T} \boldsymbol{\varepsilon}_t Q^{\pi}\left(\mathbf{s}, \mathbf{a}, t\right) \right\rangle_{w(\boldsymbol{\tau})}}{\left\langle \sum_{t=1}^{T} Q^{\pi}\left(\mathbf{s}, \mathbf{a}, t\right) \right\rangle_{w(\boldsymbol{\tau})}}.$$

until Convergence $\boldsymbol{\theta}_{k+1} \approx \boldsymbol{\theta}_k$.

In order to reduce the number of rollouts in this on-policy scenario, we reuse the rollouts through importance sampling as described in the context of reinforcement learning in [Sutton and Barto, 1998]. To avoid the fragility sometimes resulting from importance sampling in reinforcement learning, samples with very small importance weights are discarded. The expectations $E\{\cdot\}$ are replaced by the importance sampler denoted by $\langle\cdot\rangle_{w(\boldsymbol{\tau})}$. The resulting algorithm is shown in Algorithm 2.5. As we will see in Section 4, this PoWER method outperforms all other described methods significantly.

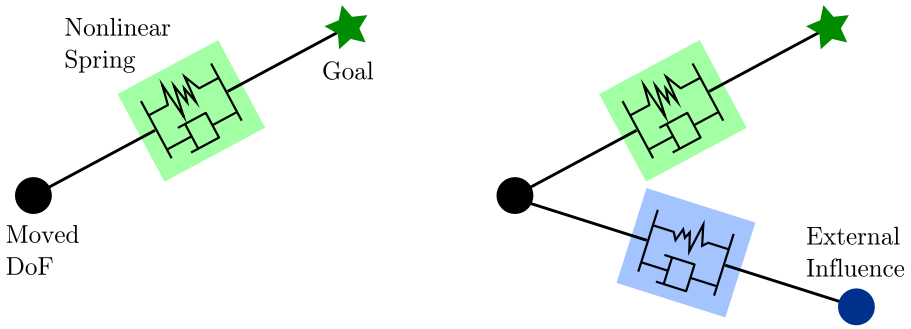
3 Motor Primitives

Motor primitives encode the elemental motions which serve as macro-actions for a higher-level supervisory system which relies on perceptual input. For example, a forehand and a backhand in tennis could be seen as two different motor primitives of the tennis player.

Biological research hindicates that humans and biological systems rely on motor primitives [Flash and Hochner, 2005, Thoroughman and Shadmehr, 2000]. For example many reflexes are still intact in a despinalized frog. It still tries to remove irritants from its skin by using precisely aimed and adapted trajectories. These trajectories even include a trajectory correction response which means that the frog is able to circumvent obstacles in the trajectory gracefully. If the grey matter of the spinal cord is stimulated, this leads to specific time variant force-fields (which correspond to motions) depending on the location of the stimulation. If these force-fields are superposed (i.e., the spinal cord is stimulated at multiple locations simultaneously) more complex motions can be achieved [Bizzi et al., 2002].

Also higher vertebrates, as rats [Tresch and Bizzi, 1999] and cats [Lemay and Grill, 2000], are organized in this way. Successive surgical lesions of a cat's central nervous system lead to less and less coordination and control. Initially the cat still has voluntary locomotion, it loses successively control, balance and coordination of all four limbs. Finally only the hind limbs have bipedal coordination [Marcoux and Rossignol, 2000]. Some execution functions and details are delegated to the lower levels of the central nervous system. These characteristics represent some kind of hierarchical architecture.

Motor primitives have also become a concept successfully used in robotics. There are several different formulations that can be grouped into two categories: based on via points or on dynamical systems. The via point based formulations basically generate the trajectories by interpolating between different via points. Most support clustering of via points belonging to several, slightly varying, demonstration trajectories. For example, Inamura et al. [2004], Williams et al. [2008] use Hidden Markov Models which encode the transition probabilities and the output probabilities. Calinon et al. [2007] use Gaussian Mixture Models [Ghahramani and Jordan, 1994], i.e., Gaussian kernels with different scales, centers and width to encode the shape of the trajectory. Drumwright et al. [2004] use exemplars [Rose et al., 1998], i.e., an interpolation between exemplary trajectories. Toussaint et al. [2007] use a sequence of attractor points in the task space based on Ijspeert's ideas. Kolter et al. [2008] use local



(a) Ijspeert Model (b) New Model

Figure 3.1: Illustration of the behavior of the motor primitives (a) and the augmented motor primitives (b).

controllers as via points and index them by space instead of using the common indexing by time. Dynamical systems use states of differential equations to generate trajectories, examples include [Ijspeert et al., 2003, Schaal et al., 2007] and [Righetti and Ijspeert, 2006]. For discrete primitives point attractor dynamical systems are used and oscillators for rhythmic primitives. The shape is modified either by a superposition of several systems or by a (nonlinear) transformation function.

For robotics, we need a suitable representation of motion as generic representation cannot cope with the high dimensionality of interesting motor systems. This representation should ideally be task-appropriate and straightforward to learn. We need smooth trajectories as jumps could damage the hardware, we would like to have a representation that has only a very limited number of parameters and those parameters should be capable of changing the speed and duration, the final state, the amplitude, etc. of the motion without changing the overall shape of the trajectory. The model by Ijspeert et al. [2003], Schaal et al. [2007] fulfills all these requirements and which we introduce in Section 3.1. It is also linear in parameters and, thus, the policy is straightforward to learn (see Section 3.2).

In this section, we first introduce the general idea behind motor primitives based on dynamical systems as suggested in [Ijspeert et al., 2002a, 2003] (Section 3.1) and, subsequently, show how perceptual coupling can be introduced. Finally, we show how the perceptual coupling can be realized by augmenting the acceleration-based framework from [Schaal et al., 2007] (Section 3.3).

3.1 Model

Figure 3.1(a) shows an illustration of the idea behind the model. In the simplest case we have a system with only one degree of freedom. We want to move this degree

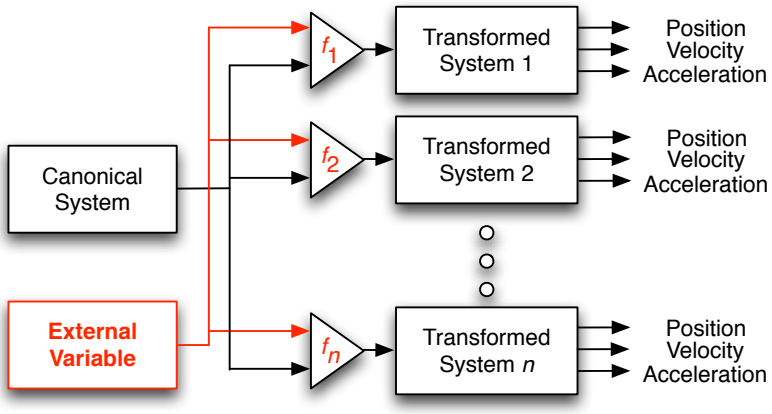


Figure 3.2: General schematic illustrating both the original motor primitive framework by [Ijspeert et al., 2003, Schaal et al., 2007] in black and the augmentation for perceptual coupling in red.

of freedom from an initial position to a final position. For example in a Swing-Up task (which we are going to use as an evaluation in Section 4.3) we have a pendulum that is hanging down and we want to swing it up into an upright position. Thus, we could imagine a spring attached to the pendulum on the one side and to the desired upright position on the other side. As we want to stabilize the pendulum in the upright position, we use a spring-damper system instead of a simple spring. In Section 4.3, we study an Underactuated Swing-Up, i.e. a system which does not have enough energy to move the pendulum directly to an upright position. In order to swing the pendulum up, it has to move in the opposite direction first in order to pump in energy. To allow more general shapes of the motion we can replace the linear spring by a nonlinear spring. See Figure 3.3 for an illustration of a motor primitive with a linear and with a nonlinear "spring". Spring-damper systems can be modeled by dynamical systems, which are used for this model.

The basic insight in the original work of Ijspeert et al. [2002a, 2003] is that motor primitives can be parted into two components, i.e., a canonical system $\mathbf{h}$ which drives transformed systems $\mathbf{g}_k$ for every considered degree of freedom k. As a result, we have system of differential equations given by

$$\dot{\mathbf{z}} = \mathbf{h}(\mathbf{z}), \tag{3.1}$$
$$\dot{\mathbf{x}} = \mathbf{g}(\mathbf{x}, \mathbf{z}, \mathbf{w}), \tag{3.2}$$

which determine the variables of internal focus $\mathbf{x}$. Here, $\mathbf{z}$ denotes the state of the canonical system and $\mathbf{w}$ the internal parameters for transforming the output of the canonical system. The schematic in Figure 3.2 illustrates this traditional setup in black.

The original formulation in [Ijspeert et al., 2002a, 2003] was a major breakthrough as the right choice of the dynamical systems in Equations (3.1, 3.2) allows determining the stability of the movement, choosing between a rhythmic and a discrete movement and is invariant under rescaling in both time and movement amplitude. With the right choice of function approximator (in our case locally-weighted regression, see Section 3.2), fast learning from a teachers presentation is possible.

While the original formulation in [Ijspeert et al., 2002a, 2003] used a second-order canonical system, it has since then been shown that a single first order system suffices [Schaal et al., 2007], i.e., we have

$$\dot{z} = h(z) = -\tau \alpha_h z,$$

which represents the phase of the trajectory. It has a time constant τ and a parameter α_h which are chosen such that the system is stable. We can now choose our internal state such that position of degree of freedom k is given by $q_k = x_{2k}$, i.e., the $2k^{\text{th}}$ component of $\mathbf{x}$, the velocity by $\dot{q}_k = \tau x_{2k+1} = \dot{x}_{2k}$ and the acceleration by $\ddot{q}_k = \tau \dot{x}_{2k+1}$. Upon these assumptions, we can express the motor primitives function $\mathbf{g}$ in the following form

$$\dot{x}_{2k+1} = \tau \alpha_g \left(\beta_g \left(t_k - x_{2k}\right) - x_{2k+1}\right) + \tau \left(\left(t_k - x_{2k}^0\right) + a_k\right) f_k, \tag{3.3}$$

$$\dot{x}_{2k} = \tau x_{2k+1}. \tag{3.4}$$

This differential equation has the same time constant τ as the canonical system, appropriately set parameters α_g, β_g, a goal parameter t_k, an amplitude modifier a_k, and a transformation function f_k. The transformation function transforms the output of the canonical system so that the transformed system can represent complex nonlinear patterns and it is given by

$$f_k(z) = \sum_{i=1}^{N} \psi_i(z)\, w_i z_k \tag{3.5}$$

where $\mathbf{w}$ are adjustable parameters and $\boldsymbol{\psi}(z)$ are weights. It uses normalized Gaussian kernels without scaling such as weights given by

$$\psi_i = \frac{\exp\left(-h_i \left(z - c_i\right)^2\right)}{\sum_{j=1}^{N} \exp\left(-h_j \left(z - c_j\right)^2\right)}. \tag{3.6}$$

These weights localize the interaction in phase space using the centers c_i and widths h_i.

Figure 3.3 shows exemplary plots for the canonical system z, the transformed system $\mathbf{x}$, the transformation function f, the weighting functions $\boldsymbol{\psi}$ and the weights $\mathbf{w}$. The length of the episode is one second. The initial state $x_1^0 = 0$ and the initial velocity

$x_0^0 = 0$. The goal parameter is $t = 1$. In the top left, the behavior of the transformed system x_1 is shown if the transformation function $f = 0$. This corresponds to the model of a linear spring pulling the system from the initial state $x_1^0 = 0$ to the goal $t = 1$ within the duration of the episode and then resting at the goal state. The canonical system z (bottom left) is decreasing from 1 to 0 during the period and resting at 0 afterwards. The shape of the policy can be changed using the transformation function f. It is defined by the weights $\mathbf{w}$ (bottom right), the canonical system z as basis function and the weighting functions $\boldsymbol{\psi}$ (bottom center). The weighting functions $\boldsymbol{\psi}$ are normalized Gaussian kernels with centers $\mathbf{c}$ and widths $\mathbf{h}$ localized in phase by the canonical system z. The resulting transformation function f is shown in the top center. The top right corner displays the transformed system x_1. This corresponds to the model of a nonlinear spring pulling the system from the initial state $x_1^0 = 0$ first in the opposite direction and then to the goal $t = 1$.

3.2 Learning for Motor Primitives

Traditionally, there are two approaches in robotics for learning policies: supervised learning and reinforcement learning. Both have their advantages and disadvantages. Supervised learning has a well defined target and it allows to learn policies fast from examples of a (human) teacher. However the policy can only reproduce the presented examples. Often this reproduction is imperfect but supervised learning cannot improve the policy without further examples. On the other hand, reinforcement learning is designed for performance related self improvement. Nevertheless, traditional reinforcement learning algorithms require exhaustive exploration of the state and action space. Our robot has seven degrees of freedom and a policy lasts for a few seconds. Due to the "curse of dimensionality" [Bellman, 1957] we cannot use a pure reinforcement learning approach as the dimensionality is prohibitively high. Thus, we need a concerted approach. We follow "nature as our teacher" and learn the policy first by imitation learning and, subsequently, improve it by reinforcement learning. This approach mimics how humans learn new motor skills. We also have a teacher explaining to us how the motion is supposed to be done or giving a demonstration. So before actually executing the motion, we have a good idea how it could be achieved. For simple tasks, this imitation often will be sufficient to perform the task successfully but for more complicated skills, we still need trial-and-error-based improvement to become successful. Furthermore performing many trials allows us to perform any task better, according to a single or combined criterion (e.g., faster, with less effort, more precisely). For example, when learning to juggle three balls we have a teacher, video or written instructions showing the motion (giving a demonstration and/or description). During the first trial, trying to reproduce the shown motion, the learning person will most likely not catch more than one ball. After a few days of practicing, most people are able to juggle continuously for a few minutes. The

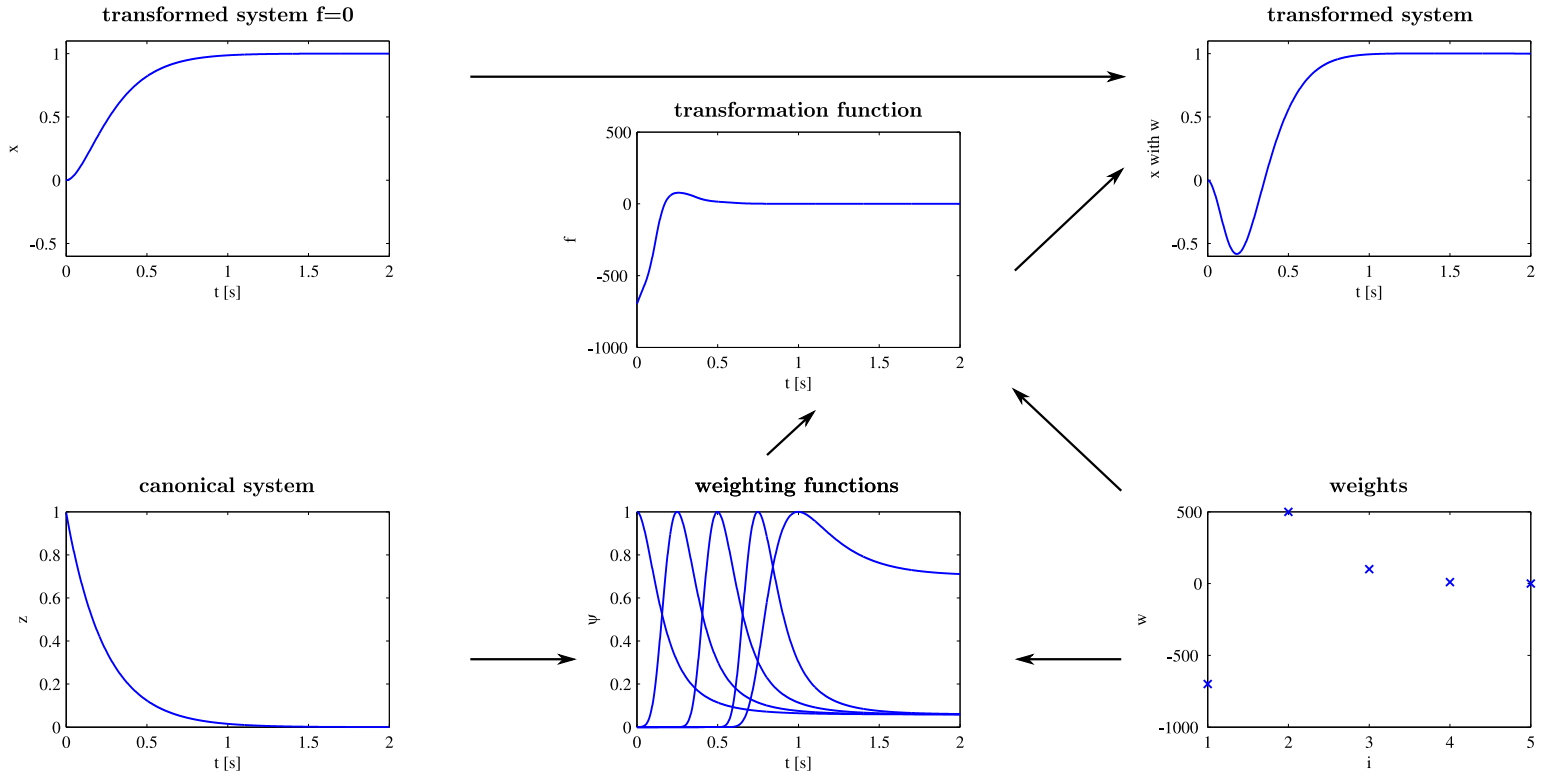


Figure 3.3: This Figure shows exemplary plots for the canonical system z, the transformed system $\mathbf{x}$, the transformation function f, the weighting functions $\boldsymbol{\psi}$ and the weights $\mathbf{w}$. See 3.1 for a detailed description.

skill can be further improved to a point where juggling can be done blindfolded.

Our concerted approach initializes the motor primitives by imitation learning. This learning is straightforward as the transformation function (3.5) of our policy representation is linear in parameters. We want to minimize the weighted squared error

$$\varepsilon_m^2 = \sum_{i=1}^{n} \psi_i^m \left(f_i^{\text{ref}} - \mathbf{z}_i^{\mathrm{T}} \mathbf{w}^m \right)^2 \tag{3.7}$$

for all parameters $\mathbf{w}^m$ with $m \in \{1, 2, \ldots, N\}$, the corresponding weighting function ψ_i^m and basis functions $\mathbf{z}_i^{\mathrm{T}}$. The reference or typical signal f_i^{ref} is the desired transformation function and i indicates the time step. Equation (3.7) can be rewritten in matrix form as

$$\varepsilon_m^2 = \left(\mathbf{f}^{\text{ref}} - \mathbf{Z}\mathbf{w}^m \right)^{\mathrm{T}} \mathbf{\Psi} \left(\mathbf{f}^{\text{ref}} - \mathbf{Z}\mathbf{w}^m \right) \tag{3.8}$$

with $\mathbf{f}^{\text{ref}}$ giving the value of f_i^{ref} for all time steps, $\mathbf{\Psi} = \operatorname{diag}(\psi_i^m, \ldots, \psi_n^m)$ and $\mathbf{Z}_i = \mathbf{z}_i^{\mathrm{T}}$. This standard optimization problem can be solved in a straightforward manner using linear regression and results in locally-weighted (linear) regression

$$\mathbf{w}^m = \left(\mathbf{Z}^{\mathrm{T}} \mathbf{\Psi} \mathbf{Z} \right)^{-1} \mathbf{\Psi}^{\mathrm{T}} \mathbf{Z} \mathbf{f}^{\text{ref}}. \tag{3.9}$$

This approach has originally been suggested by Ijspeert et al. [2002a, 2003].

After initialization of the motor primitives by locally-weighted regression for imitation learning, we use our novel algorithm PoWER (see Section 2.3) to adapt the parameters ($\mathbf{w}$, $\mathbf{t}$ and/or $\mathbf{a}$) in order to further improve the policy by reinforcement learning.

3.3 Perceptually Coupled Motor Primitives

When taking an external variable $\mathbf{y}$ into account, there are three different ways how this variable influences the motor primitive system which one can consider. The external variable could (i) only influence Equation (3.1) which would be appropriate for synchronization problems and phase-locking (similar as in [Pongas et al., 2005, Nakanishi et al., 2004a]). It could (ii) only affect Equation (3.2) which allows the continuous modification of the current state of the system by another variable. Finally, it could (iii) influence the combination of (i) and (ii).

While (i) and (iii) are the appropriate solution if phase-locking or synchronization to external triggers is needed, the coupling in the canonical system will vanquish many of the nice properties of the system and make it prohibitively hard to learn in practice. Furthermore, as we concentrate on discrete movements in this diploma thesis, we focus on the case (ii) which has not been used to date (see Figure 3.1(b)

for an illustration). In this case, we have a modified dynamical system

$$\dot{\mathbf{z}} = \mathbf{h}(\mathbf{z}), \tag{3.10}$$
$$\dot{\mathbf{x}} = \hat{\mathbf{g}}(\mathbf{x}, \mathbf{y}, \bar{\mathbf{y}}, \mathbf{z}, \mathbf{v}), \tag{3.11}$$
$$\dot{\bar{\mathbf{y}}} = \bar{\mathbf{g}}(\bar{\mathbf{y}}, \mathbf{z}, \mathbf{w}), \tag{3.12}$$

where $\mathbf{y}$ denotes the state of the external variable, $\bar{\mathbf{y}}$ the expected state of the external variable and $\dot{\bar{\mathbf{y}}}$ its derivative. This architecture inherits most positive properties from the original work while allowing the incorporation of external feedback. We will show that we can incorporate previous work with ease and that the resulting framework resembles the original work in [Schaal et al., 2007, Ijspeert et al., 2002a, 2003] while allowing to couple the external variables into the system.

In order to learn a motor primitive with perceptual coupling, we need two components. First, we need to learn the normal or average behavior $\bar{\mathbf{y}}$ of the variable of external focus $\mathbf{y}$ which can be represented by a single motor primitive $\bar{\mathbf{g}}$, i.e., we can use the same type of function from Equations (3.2, 3.12) for $\bar{\mathbf{g}}$ which are learned based on the same $\mathbf{z}$ and given by Equations (3.3, 3.4). Additionally, we have the system $\hat{\mathbf{g}}$ for the variable of internal focus $\mathbf{x}$ which determines our actual movements incorporating the inputs of the normal behavior $\bar{\mathbf{y}}$ as well as the current state $\mathbf{y}$ of the external variable. We obtain the system $\hat{\mathbf{g}}$ by inserting a modified coupling function $\hat{\mathbf{f}}(\mathbf{z}, \mathbf{y}, \bar{\mathbf{y}})$ instead of the original $\mathbf{f}(z)$ in $\mathbf{g}$. This new function $\hat{\mathbf{f}}(z, \mathbf{y}, \bar{\mathbf{y}})$ is augmented in order to include perceptual coupling to $\mathbf{y}$ and we obtain

$$\hat{f}_k(z, \mathbf{y}, \bar{\mathbf{y}}) = \sum_{i=1}^{N} \psi_i(z)\, \hat{w}_i z_k + \sum_{j=1}^{M} \hat{\psi}_j(z) \left(\boldsymbol{\kappa}_{jk}^{\mathrm{T}} (\mathbf{y} - \bar{\mathbf{y}}) + \boldsymbol{\delta}_{jk}^{\mathrm{T}} \left(\dot{\mathbf{y}} - \dot{\bar{\mathbf{y}}} \right) \right), \tag{3.13}$$

where $\hat{\psi}_j(z)$ denotes Gaussian kernels as in Equation (3.6) with centers $\hat{c}_j$ and width $\hat{h}_j$. Note, that it can be useful to set $N > M$ to reduce the number of parameters. All parameters are given by $\mathbf{v} = [\hat{\mathbf{w}}, \boldsymbol{\kappa}, \boldsymbol{\delta}]$. Here, $\hat{\mathbf{w}}$ are just the standard transformation parameters while $\boldsymbol{\kappa}_{jk}$ and $\boldsymbol{\delta}_{jk}$ are the local coupling factors which can be interpreted as gains acting on the difference between the desired behavior of the external variable and its actual behavior. Note that for noise-free behavior and perfect initial positions such coupling would never play a role. Thus, the approach would simply become the original approach suggested in [Ijspeert et al., 2002a, 2003] and presented in Section 3.1. However, in the noisy, imperfect case, this perceptual coupling can ensure success even in extreme cases.

3.4 Learning for Perceptually Coupled Motor Primitives

While the transformation function $\mathbf{f}_k(\mathbf{z})$ can be learned from few presentations or even just a single example, this simplicity no longer transfers to learning the new

function $\hat{f}_k(z, \mathbf{y}, \bar{\mathbf{y}})$ as perceptual coupling requires the coupling to an uncertain external variable. While imitation learning approaches are feasible, they require larger numbers of presentations of a teacher with very similar kinematics for learning the behavior sufficiently well. It is also hard to find consistent examples. As an alternative, we use a similar approach of imitation and self-improvement by trial-and-error as in Section 3.2.

For imitation learning, we can largely follow the original work in [Ijspeert et al., 2002a, 2003] and only need minor modifications to incorporate the perceptual coupling from Equation (3.13). We also make use of locally-weighted regression in order to determine the optimal motor primitives, use the same weighting and compute the targets based on the dynamical systems. However, unlike in [Ijspeert et al., 2002a, 2003], we need a bootstrapping step as we determine the parameters first for the system described by Equation (3.12) and, subsequently, use the learned results in the learning of the system in Equation (3.11). For doing so, we can compute the regression targets for the first system by taking Equation (3.3), replacing $\bar{\mathbf{y}}$ and $\dot{\bar{\mathbf{y}}}$ by samples of $\mathbf{y}$ and $\dot{\mathbf{y}}$, and solving for $f_k(z)$ as discussed in [Ijspeert et al., 2002a, 2003] and Section 3.2. A local regression yields appropriate values for the parameters of $f_k(z)$. Subsequently, we can perform the exact same step for $\hat{f}_k(z, \mathbf{y}, \bar{\mathbf{y}})$ where only the number of variables has increased but the resulting regression follows analogously. However, note that while a single demonstration suffices for the parameter vector $\mathbf{w}$ and $\hat{\mathbf{w}}$, the parameters $\boldsymbol{\kappa}$ and $\boldsymbol{\delta}$ cannot be learned by imitation as these require deviation from the nominal behavior for the external variable.

However, as discussed before, pure imitation of perceptual coupling can be difficult for learning the coupling parameters as well as the best nominal behavior for a robot with kinematics different from the human, many different initial conditions and in the presence of significant noise. Thus, we suggest to improve the policy by trial-and-error using reinforcement learning upon an initial imitation. As the reward achieved by perceptual coupling depends not only on the fitness of the parameters $\boldsymbol{\kappa}$ and $\boldsymbol{\delta}$ but also on the perturbations, we average over the rewards of a small batch of rollouts in order not to optimize for one specific kind of perturbation. The problem is much harder as we have to deal with a variety of perturbations and, thus, convergence is slower.

4 Evaluations

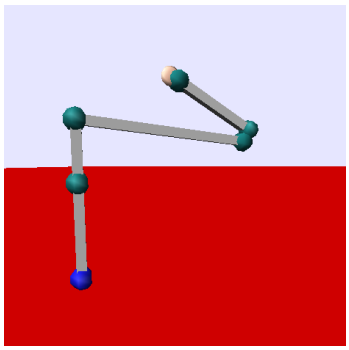
In this section, we demonstrate the effectiveness of the algorithm presented in Section 2.3 in the context of motor primitive learning for robotics as well as the effectiveness of the augmented framework for perceptually coupled motor primitives as presented in Section 3.3.

As first evaluation, we will show that the novel presented PoWER algorithm outperforms all previous well-known methods, i.e., Finite Difference Gradients [Tedrake et al., 2004, Peters and Schaal, 2006b, Kohl and Stone, 2004], 'Vanilla' Policy Gradients [Williams, 1992, Sutton et al., 2000, Lawrence et al., 2003, Tedrake et al., 2004, Baxter et al., 2001, Baxter and Bartlett, 2001, Peters and Schaal, 2006b], the Episodic Natural Actor Critic [Bagnell and Schneider, 2003, Peters et al., 2003a, Peters and Schaal, 2006b] and the generalized Reward-Weighted Regression [Peters and Schaal, 2007b] on the two simulated benchmark problems suggested in [Peters and Schaal, 2006b] and a simulated Underactuated Swing-Up [Atkeson, 1994].

Real robot applications are done with our best benchmarked method, the PoWER method. Here, we first show PoWER can learn the Underactuated Swing-Up [Atkeson, 1994] even on a real robot. As a significantly more complex motor learning task, we show that this method can be used in learning a complex, high-speed, real-life motor learning problem Ball-in-a-Cup [Wikipedia, 2008a] with motor primitives for all seven degrees of freedom of our Barrett WAM$^{\text{TM}}$ robot arm. Finally, we show that we can learn perceptual coupling for Ball-in-a-Cup in a physically realistic simulation of an anthropomorphic robot arm. Ball-in-a-Cup is a good benchmark for testing the motor learning performance as it is sufficiently complex for humans. We show that we can learn the problem roughly at the efficiency of a young child. PoWER successfully creates a perceptual coupling which results in robustness even to perturbations that are very challenging for a skilled adult player.

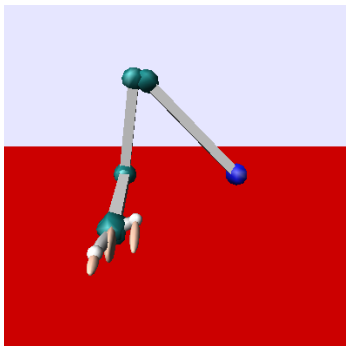
4.1 Experimental Setup

The evaluations were done in simulation as well as on a real physical robot. In Section 4.1.1 we introduce the hardware and software we used and in Section 4.1.2 we describe how Ball-in-a-Cup was modelled for simulation.



(a) Simulated Barrett WAM™ Arm

(b) Real Barrett WAM™ Arm



(c) Simulated SARCOS Master Arm

Figure 4.1: This Figure shows the robots that were used for the evaluations.

4.1.1 Hardware and Software

The benchmark comparison in Section 4.2 was done using MATLAB™. The motor primitives are programmed as structures containing the parameters and the current state. The dynamical systems are solved by numerical integration using the semi-implicit Euler algorithm. The implementation of the presented reinforcement learning algorithms was done using only standard MATLAB™ commands. We compared the results of these algorithms to the result of the Optimization Toolbox.

We used a simulated Barrett WAM™ Arm and a real Barrett WAM™ Arm for the Underactuated Swing-Up as well as for Ball-in-a-Cup. For the evaluation of the perceptual coupling we used a simulated SARCOS Master Arm (see Figure 4.1).

Both the Barrett WAM™ Arm and the SARCOS Master Arm are anthropomorphic seven degrees of freedom arms. The Barrett WAM™ Arm uses differential cable drives and has the heavy motors located in the base. As a result the arm is compliant, backdriveable and is capable of high speed motions. The controller for the joint positions and velocities is model based and runs at 500 Hz on a real-time Linux computer using Xenomai. The desired motor torques are sent to the robot via a CAN bus. This bus also transfers the signals from the joint encoders. The motor currents are controlled onboard within the robot. The motor primitives yield the desired positions, velocities and accelerations which are cascaded through the joint controllers and the motor controllers.

As simulation environment, we use SL [Schaal, 2004]. The program is split in different servos, including the task servo, the vision servo, the motor servo, the simulation servo and the robot servo. This modular design makes it possible to switch between a physically realistic simulation and controlling the real robot straightforwardly as the simulation servo simply has to be switched to the robot servo, and the vision

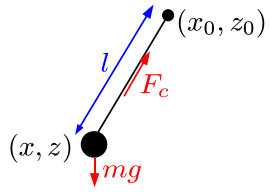


Figure 4.2: This figure introduces the used notations: (x, z) is the position of the ball, (x_0, z_0) is the position where the string is attached to the cup. Gravity mg pulls the ball downwards (where m is the mass of the ball and $g = 9.81\text{m/s}^2$ the gravitational constant), F_c pulls the ball towards the cup. The current string length is denoted by l, the nominal string length by l_0.

servo from a simulated one to one getting data from real cameras. The work in this diploma thesis consisted of programming the task servo, which handles the generation of the desired state of the robot, based on the information from the vision servo and proprioceptive feedback from the robot joints (via the simulation servo or the robot servo). The desired state is passed to the motor servo, which outputs motor torques. These motor torques are then passed either to the simulation servo or the real robot. The other servos were built by other members of our group. The vision servo is based on two cameras using a blob detector and a learned calibration to the robot coordinate system to get the position of the ball for the Ball-in-a-Cup task. SL is written in C, uses OpenGL for display in the state of simulation and it runs on Unix and Unix-like operating systems.

4.1.2 Modeling Ball-in-a-Cup

The simulation servo in SL only simulates the robot itself and interactions with simple objects. Thus, we need to model Ball-in-a-Cup by hand.

The model is based on a point mass attached to a point by a spring-damper system. The spring-damper system models an elastic string at nominal string length and the forces if the string is stretched even further. If the distance of the ball to the cup is less than the nominal string length the ball is simulated as a free point mass.

Constrained Dynamics. In order to obtain the constrained dynamics, we employed Gauss' principle [Udwadia and Kalaba, 1996]. For better readability, we show the derivation in two dimensions. It is straightforward to fill in the missing terms for the third dimension. See Figure 4.2 for the definitions of the variables.

The forces acting on the ball can be written as

$$\begin{aligned} m\ddot{x} &= F_c^x, \\ m\ddot{z} &= F_c^z - mg. \end{aligned} \tag{4.1}$$

The string is modeled as a spring damper system given by

$$\begin{aligned} \ddot{l} + d\dot{l} + k\,(l - l_0) &= 0, \\ \ddot{l} &= -d\dot{l} - k\,(l - l_0)\,, \end{aligned} \tag{4.2}$$

where d is the dampening constant and k the spring constant. The current length of the string is given by the Pythagorean theorem

$$l = (x - x_o)^2 + (z - z_o)^2 . \tag{4.3}$$

Differentiating this length (4.3) twice leads to

$$\begin{aligned} \dot{l} &= 2(x - x_o)(\dot{x} - \dot{x}_o) + 2(z - z_o)(\dot{z} - \dot{z}_o) && (4.4) \\ \ddot{l} &= 2(\dot{x} - \dot{x}_o)^2 + 2(x - x_o)(\ddot{x} - \ddot{x}_o) \\ &+ 2(\dot{z} - \dot{z}_o)^2 + 2(z - z_o)(\ddot{z} - \ddot{z}_o) . && (4.5) \end{aligned}$$

If we insert Equations (4.3, 4.4) into Equation (4.2), then insert this result in Equation (4.7) and rearrange the expression, we get

$$\underbrace{\begin{bmatrix} (x - x_0) & (z - z_0) \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} \ddot{x} \\ \ddot{z} \end{bmatrix} = b \tag{4.6}$$

with

$$\begin{aligned} b &= -(\dot{x} - \dot{x}_o)^2 - (\dot{z} - \dot{z}_o)^2 + (x - x_o)\ddot{x}_o + (z - z_o)\ddot{z}_o \\ &- d\left((x - x_o)(\dot{x} - \dot{x}_o) + (z - z_o)(\dot{z} - \dot{z}_o)\right) \\ &- \frac{k}{2}\left((x - x_o)^2 + (z - z_o)^2 - l_0\right) . \end{aligned} \tag{4.7}$$

For an equation of the form $\mathbf{A}\ddot{\mathbf{x}} = \mathbf{b}$, Gauss' Principle can be applied yielding the constraint forces

$$\mathbf{F}_c = \mathbf{M}^{\frac{1}{2}} \left(\mathbf{A}\mathbf{M}^{-\frac{1}{2}}\right)^{+} \left(\mathbf{b} - \mathbf{A}\mathbf{M}^{-1}\mathbf{F}\right) \tag{4.8}$$

with the pseudo inverse $\mathbf{A}^{+} = \left(\mathbf{A}\mathbf{A}^{\mathrm{T}}\right)^{-1}\mathbf{A}^{\mathrm{T}}$, the matrix root $\mathbf{M}^{\frac{1}{2}}\mathbf{M}^{\frac{1}{2}} = \mathbf{M}$, the generalized mass matrix $\mathbf{M}$ and the internal forces $\mathbf{F}$ (i.e., all forces not included in the constrained forces $\mathbf{F}_c$). Inserting Equation (4.6), $\mathbf{M} = m\mathbf{I}$ (where I is the identity matrix) and $\mathbf{F} = [0, -mg]^{\mathrm{T}}$ in Equation (4.8), we obtain

$$\begin{aligned} \mathbf{F}_c &= m\left(\mathbf{A}\mathbf{A}^{\mathrm{T}}\right)\mathbf{A}^{\mathrm{T}}\left(b - \frac{1}{m}\mathbf{A}\mathbf{F}\right), \\ &= \frac{m}{(x - x_o)^2 + (z - z_o)^2} \begin{bmatrix} x - x_0 \\ z - z_0 \end{bmatrix} \left(b + \frac{m}{m}(z - z_0)\, g\right) . \end{aligned} \tag{4.9}$$

Using Equation (4.7) we get the constraint forces

$$\mathbf{F}_c = F_c \frac{1}{(x - x_o)^2 + (z - z_o)^2} \begin{bmatrix} x - x_0 \\ z - z_0 \end{bmatrix} \tag{4.10}$$

(a) This plot shows a swinging motion; the simulation reflects the real behavior perfectly.

(b) In this plot we see an actual rollout; the recorded ball hits the rim of the cup, the simulated ball is a few millimeters off and does fly past the cup, thus, the the different behavior in the last third of the rollout.

Figure 4.3: These plots show the behavior of the simulated Ball-in-a-Cup in comparison to the captured data. The cup movements and initial conditions are taken from the recorded data.

with

$$\begin{aligned} F_c &= m\Big(-(\dot{x}-\dot{x}_o)^2-(\dot{z}-\dot{z}_o)^2+(x-x_o)\,\ddot{x}_o+(z-z_o)\,\ddot{z}_o \\ &- d\left((x-x_o)\,(\dot{x}-\dot{x}_o)+(z-z_o)\,(\dot{z}-\dot{z}_o)\right) \\ &- \frac{k}{2}\Big((x-x_o)^2+(z-z_o)^2-l_0\Big)+(z-z_0)\,g\Big). \end{aligned} \tag{4.11}$$

Free Dynamics. If the distance between the ball and the cup is less than the string length, the ball is modeled as a free flying point mass with $\ddot{x}=0$ and $\ddot{z}=-g$.

Collisions and Reflections: When collisions of ball and cup are detected the accelerations are adjusted accordingly. As result

$$\ddot{b}=\frac{\vartheta\left(\dot{b}-\dot{c}\right)}{\Delta t} \tag{4.12}$$

would be the new acceleration of the ball with velocity $\dot{b}$ after reflection on a plane of the cup with velocity $\dot{c}$. For a perfect reflection $\vartheta=2$. We used $\vartheta=1.5$ to model the energy loss. As the ball is a sphere, the contact plane of the ball is always orthogonal to the other contact plane. If we transform the system to "Cup-Space", it is easy to find these planes either directly or using the intercept theorem.

Numerical Simulation. As SL is written in C the equations were also implemented in C. The numerical simulation runs at 500Hz. For the integrations we use the semi-implicit Euler algorithm (also called symplectic Euler, semi-explicit Euler, Euler–Cromer, and Newton–Størmer–Verlet [Wikipedia, 2008b])

$$\begin{aligned} \dot{x}^i &= \dot{x}^{i-1}+\Delta t\ddot{x}^{i-1} \\ x^i &= x^{i-1}+\Delta t\dot{x}^i \end{aligned} \tag{4.13}$$

with time index i and time step $\Delta t=0.002$s. This combination of an explicit and implicit Euler step yields better results in comparison to the explicit Euler as the energy is almost conserved, while implementation complexity and calculational costs are the same [Verlet, 1967, Vesely, 2001, Giordano and Nakanishi, 2005].

Verification. Simulation with initial conditions and cup trajectories taken from motion captured data, confirmed that our simulation is physically realistic (see Figure 4.3) once the spring and damper coefficients were adjusted to reflect the behavior of the real system. For motion capturing, we used a VICON$^{\text{TM}}$ setup (see Figure 4.4). Our setup has twelve cameras in total, three in each corner. The cameras emit infrared light which is reflected on passive infrared reflective markers. The supplied software is used to track the markers in the two dimensional camera images and fuse the information of the twelve cameras for the three dimensional Cartesian coordinates of the markers. The human performer wears a shirt with 23

(a) Close-up of a marker.

(b) Cameras.

(c) Ball-in-a-Cup with markers.

(d) Human demonstrator performing Ball-in-a-Cup wearing a shirt with markers.

Figure 4.4: The VICON^{TM} motion capture setup is based on passive infrared-reflective markers and active infrared cameras. Our setup uses twelve cameras in total.

markers. The software has a model of the human upper body with the corresponding markers. The software fits the parts of the body to the markers, taking into account constraints defined in the model. The software also calculates the positions of the body parts (which do not necessarily correspond directly to the markers) and the joint angles. The cup has five markers attached to it in order to obtain the Cartesian position and rotations. The ball is a marker itself and only the Cartesian position can be obtained. For the verification of the Ball-in-a-Cup simulation we used the Cartesian coordinates of the cup and the ball. The Cartesian positions, velocities and accelerations of the cup were "played back" in the simulator and we verified that the simulated ball behaves the same way as the motion-captured ball, if we set the simulated initial conditions to the real ones. For Section 4.5 we also used the joint angles, velocities and accelerations of the human performer's right arm.

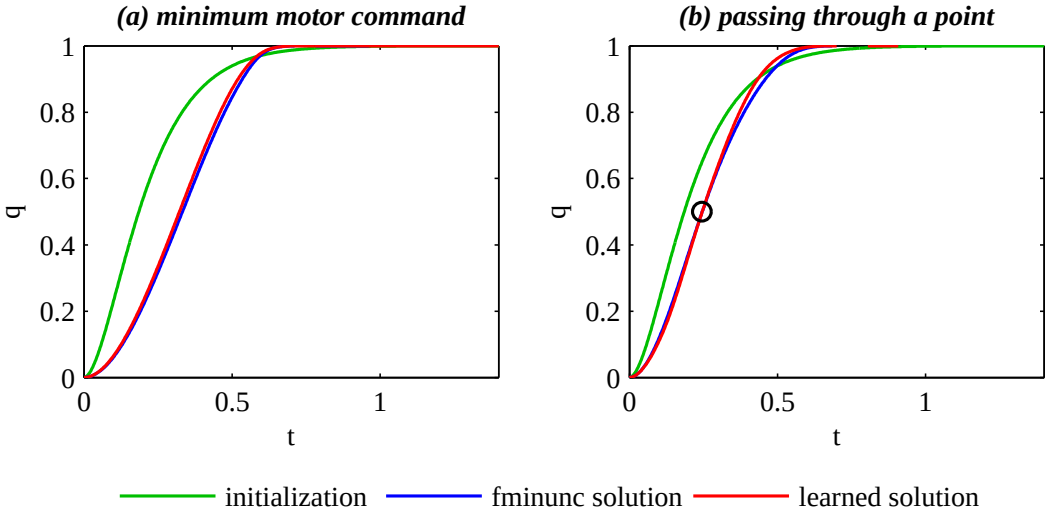


Figure 4.5: This figure shows the solutions for the benchmark problems, $t = 1.4$ corresponds to T. To verify the found solutions we used the MATLAB^{TM} Optimization Toolbox (`fminunc`). The learned solutions only differ very slightly from the optimal ones (for the given reward and the limited representational power of the motor primitives).

4.2 Benchmark Comparison

As benchmark comparison, we intend to follow a previously studied scenario in order to evaluate which method is best-suited for our problem class. For doing so, we perform our evaluations on the same benchmark problems as [Peters and Schaal, 2006b] and use two tasks commonly studied in motor control literature [Ben-Itzhak and Karniel, 2008] for which the analytic solutions are known. In this comparison, we mainly want to show the suitability of our algorithm and show that it outperforms previous methods such as Finite Difference Gradient (FDG) methods [Tedrake et al., 2004, Peters and Schaal, 2006b, Kohl and Stone, 2004], 'Vanilla' Policy Gradients (VPG) with optimal baselines [Williams, 1992, Sutton et al., 2000, Lawrence et al., 2003, Tedrake et al., 2004, Baxter et al., 2001, Baxter and Bartlett, 2001, Peters and Schaal, 2006b], the Episodic Natural Actor Critic (eNAC) [Bagnell and Schneider, 2003, Peters et al., 2003a, Peters and Schaal, 2006b], and the episodic version of the Reward-Weighted Regression (RWR) algorithm [Peters and Schaal, 2007b].

We consider two standard tasks taken from [Peters and Schaal, 2006b] but we use the newer form of the motor primitives from [Schaal et al., 2007]. The first task is to achieve a goal with a minimum-squared movement acceleration and a given

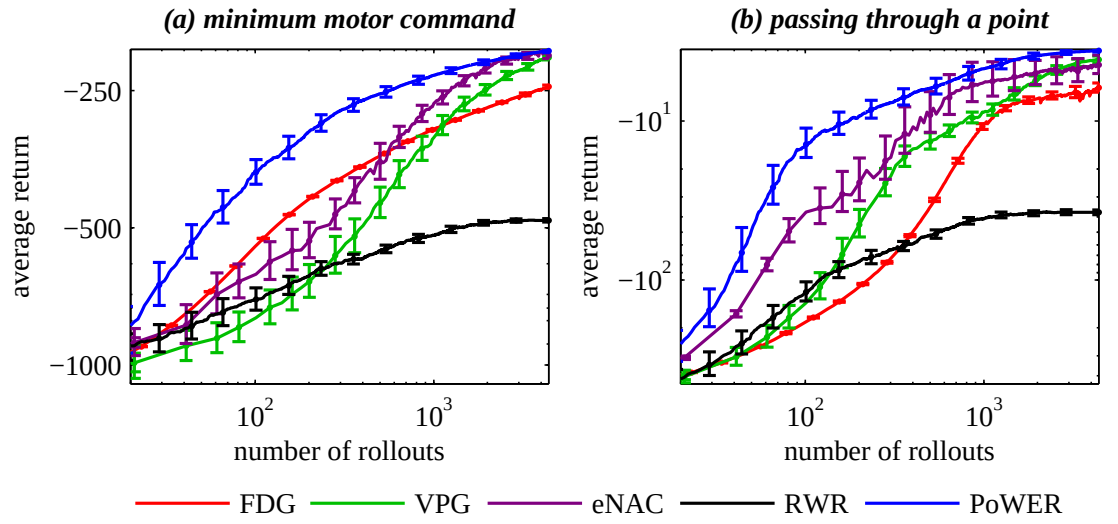


Figure 4.6: This figure shows the mean performance of all compared methods in two benchmark tasks averaged over twenty learning runs with the error bars indicating the standard deviation. Policy learning by Weighting Exploration with the Returns (PoWER) clearly outperforms Finite Difference Gradients (FDG), 'Vanilla' Policy Gradients (VPG), the Episodic Natural Actor Critic (eNAC) and the adapted Reward-Weighted Regression (RWR) for both tasks.

movement duration, i.e. a return of

$$R(\boldsymbol{\tau}) = -\sum_{i=0}^{T/2} c_1 \ddot{q}_i^2 - \sum_{i=\frac{T}{2}+1}^{T} c_2 \left[(q_i - g)^2 + \dot{q}_i^2 \right] \tag{4.14}$$

is being optimized, where $c_1 = 1/100$ is the weight of the transient rewards for the movement duration $T/2$, while $c_2 = 1000$ is the importance of the final reward, extended over the time interval $[T/2 + 1, T]$, which insures, that the goal state $g = 1.0$ is reached and maintained properly (see Figure 4.5a). The initial state of the motor primitive is always zero in this toy evaluation.

The second task involves passing through an intermediate point during the trajectory, while minimizing the squared accelerations, i.e., we have a similar return with an additional punishment term for missing the intermediate point p_M at time M given by

$$R(\boldsymbol{\tau}) = -\sum_{i=0}^{T/2} \widetilde{c}_1 \ddot{q}_i^2 - \sum_{i=\frac{T}{2}+1}^{T} \widetilde{c}_2 \left[(q_i - g)^2 + \dot{q}_i^2 \right] - \widetilde{c}_3 (q_M - p_M)^2 \tag{4.15}$$

where $\widetilde{c}_1 = 1/10000$, $\widetilde{c}_2 = 200$, $\widetilde{c}_3 = 20000$. The goal has a value of $g = 1.0$, the intermediate point a value of $p_M = 0.5$ at time $M = 7/40T$ and the initial state was

zero. This return yields a smooth movement which passes through the intermediate point before reaching the goal, even though the optimal solution is not identical to the analytic solution with hard constraints (see Figure 4.5b).

All open parameters were manually optimized for each algorithm in order to maximize the performance while not destabilizing the convergence of the learning process. When applied in the episodic scenario, Policy learning by Weighting Exploration with the Returns (PoWER) clearly outperforms the Episodic Natural Actor Critic (eNAC), 'Vanilla' Policy Gradient (VPG), Finite Difference Gradient (FDG) and the adapted Reward-Weighted Regression (RWR) for both tasks. The episodic Reward-Weighted Regression (RWR) is outperformed by all other algorithms suggesting that this algorithm does not generalize well from the immediate reward case. While FDG gets stuck on a plateau, both eNAC and VPG converge to the same, good final solution. PoWER finds the same (or even slightly better) solution while achieving it noticeably faster. The results are presented in Figure 4.6. Note that this plot has logarithmic scales on both axes, thus, a unit difference corresponds to an order of magnitude. The omission of the first twenty rollouts was necessary to cope with the log-log presentation.

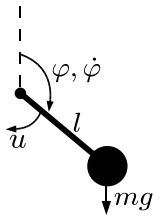


Figure 4.7: This figure shows a schematic drawing of the pendulum. Gravity $g = 9.81\text{m/s}^2$ is pulling the pendulum of mass $m = 0.5\text{kg}$ and length $l = 0.6\text{m}$ downwards. Additionally the motor torque u is applied to the pendulum. The combined forces of gravity and motor torque move the pendulum and yield the angle $\varphi \in [-\pi, \pi]$ and the angular velocity $\dot{\varphi}$.

4.3 Underactuated Swing-Up

As additional simulated benchmark and for the real-robot evaluations, we employed the Underactuated Swing-Up [Atkeson, 1994]. Here, only a single degree of freedom of the robot is used. This single degree of freedom is represented by one motor primitive as described in Section 3.1. The goal of the task is to move a heavy pendulum that is hanging downward to an upright position and stabilize it there. See Figure 4.7 for an illustration. The system can be modeled as

$$ml^2\ddot{\varphi} = -\mu\dot{\varphi} + mgl\sin\varphi + u \tag{4.16}$$

with angles $\varphi \in [-\pi, \pi]$, torque u, mass m, gravity g and length of the pendulum l. The objective is threefold: the pendulum has to be swung up in minimum time, has to be stabilized in the upright position and, at the same time, do this with minimal motor torques. By limiting the motor current for that degree of freedom, we can ensure that the torque limits $|u| \leq u_{\max}$ described in [Atkeson, 1994] are maintained and directly moving the joint to the upright position is not possible. Under these torque limits, the robot needs to (a) first move away from the target to limit the maximal required torque during the swing-up in (b-d) and subsequent stabilization (e) as illustrated in Figure 4.9(a-e). This problem is similar to a mountain-car problem, a frequently used benchmark in the reinforcement learning literature. In this problem we have a car placed in a valley, it is supposed to go on the top of the mountain but does not have the necessary capabilities of acceleration to do so directly. Thus, the car has to drive up the mountain on the opposite side of the valley first, in order to gain sufficient energy. The standard mountain-car problem is designed to get the car to the top of the mountain in minimum time, if it stops at this point or drives with high speed does not matter, usage of the accelerator and break is not punished. Adding the requirement of stabilizing the car at the top of the mountain makes the problem significantly harder. The Underactuated Swing-Up considers these additional constraints and requires the car to stop on top or experience a failure.

The applied torque limits were the same as in [Atkeson, 1994] and so was the reward function except that the complete return of the trajectory was transformed by an

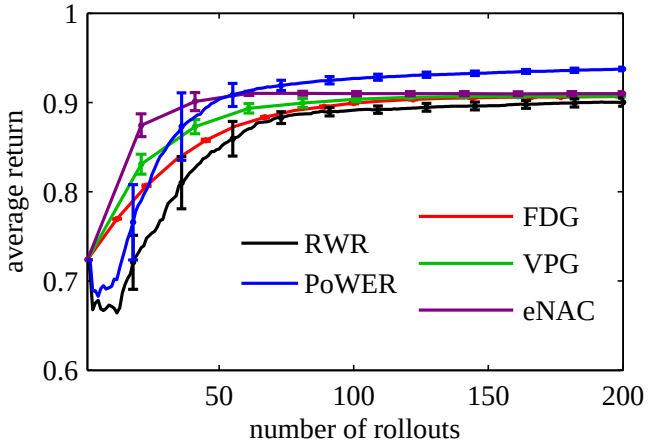


Figure 4.8: This figure shows the performance of all compared methods for the swing-up in simulation and show the mean performance averaged over 20 learning runs with the error bars indicating the standard deviation. PoWER outperforms the other algorithms from 50 rollouts on and finds a significantly better policy.

$\exp(\cdot)$ to ensure positivity. The reward function is

$$r = \alpha \left(\frac{\varphi}{\pi}\right)^2 + \beta \left(\frac{2}{\pi}\right)^2 \log\cos\left(\frac{\pi}{2}\frac{u}{u_{\max}}\right). \tag{4.17}$$

The first term of the sum is punishing the distance to the desired upright position and the second term punishing the usage of motor torques. Again all open parameters were manually optimized. The motor primitive with nine shape parameters and one goal parameter was initialized by imitation learning from a kinesthetic teach-in. As the pendulum is attached to the outer link of the robot we can simply modify the mass and inertia data for the outer link in SL to obtain a simulation of the pendulum. Subsequently, we compared the other algorithms as previously considered in Section 4.2 and could show that PoWER would again outperform all other methods. The results are given in Figure 4.8. As it turned out to be the best performing method, we then used it successfully for learning optimal swing-ups on a real robot. See Figure 4.9(f) and Figure 4.10 for the resulting real-robot performance.

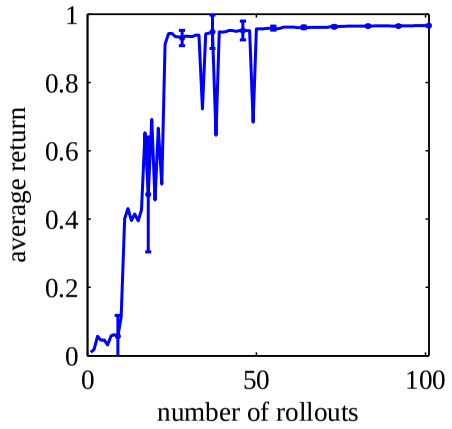
(a) Move away from the target to limit the maximal required torque during the swing-up.

(b) Swing-up.

(c) Swing-up.

(d) Swing-up.

(e) Stabilization.



(f) The performance of the PoWER method on the real robot.

Figure 4.9: This figure shows the time series of the Underactuated Swing-Up where only a single joint of the robot is moved with a torque limit ensured by limiting the maximal motor current of that joint.

(a) Initial policy after imitation learning (without torque limit).

(b) Initial policy after imitation learning (with active torque limit).

(c) Policy after 20 rollouts, going further up.

(d) Policy after 30 rollouts, going too far.

(e) Policy after 40 rollouts, going only a bit too far.

(f) Final policy after 65 rollouts.

Figure 4.10: This figure shows the improvement of the policy over rollouts. The snapshots from the video show the final positions.

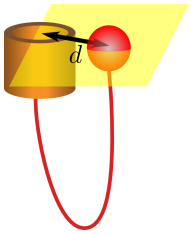


Figure 4.11: This figure illustrates how the reward is calculated. The yellow plane represents the level of the upper rim of the cup. For a successful rollout the ball has to be moved above the cup first and is than flying in a downward direction into the opening of the cup. The reward is calculated as the distance d of the center of the cup and the center of the ball on the plane at the moment the ball is passing the plane in a downward direction. If the ball is flying directly in the center of the cup, the distance is 0 and through the transformation $\exp(-d)$ yields the highest possible reward of 1. The further the ball passes the plane from the cup, the larger the distance and, thus, the smaller the resulting reward.

4.4 Learning Ball-in-a-Cup

The most challenging application in this evaluation is the children game Ball-in-a-Cup, also known as Balero, Bilboquet or Kendama [Wikipedia, 2008a]. There are two versions of this game. The first one is called Ball-in-a-Cup in the United States of America or Fangbecher in Germany. This version, which we are going to use, has a small cup which is held in one hand (or attached at the robot's end-effector) and the cup has a small wooden ball hanging down from it on a string (40cm for our toy). Initially, the ball is hanging down vertically. The player needs to move fast in order to induce a motion at the ball through the string, toss it up and catch it with the cup, a possible movement is illustrated in Figure 4.14 (top row). The alternative version is called Bilboquet in France; Balero, Boliche, Emboque, Perinola or Juego de la Cocoa in Spanish and Portuguese speaking countries such as Argentina, Ecuador, Colombia, Mexico, Spain, Chile, Brazil or Venezuela. Here the ball is larger and has a hole. It is attached to a stick via a string. The goal is to throw the ball up and catch it in such a way, that the ball is "impaled" on the stick. Historical traces in Europe lead back to France in the sixteenth century where this game was esteemed by King Henry III. The game had two big revivals in France, one in the eighteenth century under Louis XV and another one in 1910. The game was not originally a children's game. There exist many variations on the size and shape of the toy. Kendama, a traditional Japanese toy, is somewhat a combination of the two versions. Here the player has several locations where the ball can be caught: either on one of the three dishes of various sizes, on the spike or somewhere on the crossing. Advanced players perform tricks and combinations and there are Kendama championships. See Figure 4.12 for photos of the various toys.

The state of the system is described in joint angles and velocities of the robot and the Cartesian coordinates of the ball. The actions are the joint space accelerations where each of the seven joints is represented by a motor primitive. All motor primitives

(a) Ball-in-a-Cup

(b) Fangbecher

(c) Balero

(d) Bilboquet

(e) Kendama

Figure 4.12: This figure shows photos of the different toys. http://www.toypost.co.uk, http://www.kendama.jp, http://en.wikipedia.org, http://commons.wikimedia.org

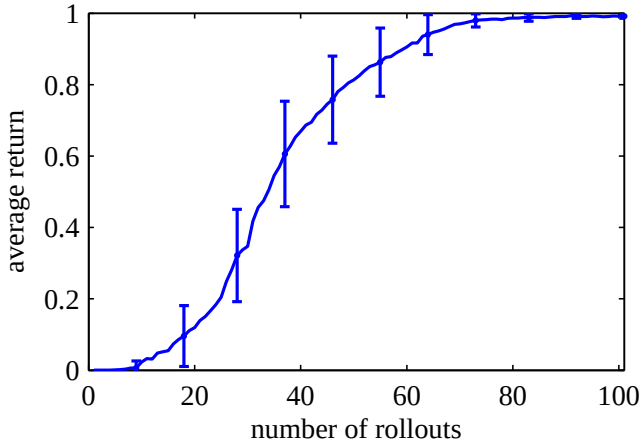


Figure 4.13: This figure shows the expected return of the learned policy in the Ball-in-a-Cup evaluation averaged over 20 runs.

are perturbed separately but employ the same joint final reward given by

$$r(t) = \begin{cases} \exp\left(-\alpha\left(x_c - x_b\right)^2 - \alpha\left(y_c - y_b\right)^2\right) & \text{if } t = t_c \\ 0 & \text{else} \end{cases} \tag{4.18}$$

where t_c is the moment where the ball passes the rim of the cup with a downward direction, the cup position denoted by $[x_c, y_c, z_c] \in \mathbb{R}^3$, the ball position $[x_b, y_b, z_b] \in \mathbb{R}^3$ and a scaling parameter $\alpha = 100$ (also see Figure 4.11). The task is quite complex as the reward is not modified solely by the movements of the cup but foremost by the movements of the ball which are very sensitive to changes in the movement. A small perturbation of the initial condition or during the trajectory will drastically change the movement of the ball and hence the outcome of the rollout. In order to cancel these perturbations, we need perceptual coupling which we learn in Section 4.5.

Due to the complexity of the task, Ball-in-a-Cup is even a hard motor task for children who only succeed at it by observing another person playing[1]. Subsequently, a lot of improvement by trial-and-error is required until the desired solution can be achieved in practice. The child will have an initial success as the initial conditions and executed cup trajectory fit together by chance, afterwards the child still has to practice a lot until it is able to get the ball in the cup (almost) every time and so cancel various perturbations. Learning the necessary perceptual coupling to get the ball in the cup (almost) every time is even a hard task for adults, as we have verified by self-experimentation and with other people from our department. In contrast to a

[1] or deducing from similar previously learned tasks how to maneuver the ball above the cup in such a way that it can be caught. Humans learn a general hand-eye coordination as well as throwing and catching strategies, that can be applied to a wide variety of tasks including Ball-in-a-Cup.

tennis swing, where a human just needs to learn a goal function for the one moment the racket hits the ball, in Ball-in-a-Cup we need a complete dynamical system as cup and ball constantly interact. Mimicking how children learn to play Ball-in-a-Cup, we first initialize the motor primitives by imitation and, subsequently, improve them by reinforcement learning in order to get an initial success (this Section). Afterwards, we also acquire the perceptual coupling by reinforcement learning (Section 4.5).

We recorded the motions of a human player by kinesthetic teach-in in order to obtain an example for imitation as shown in Figure 4.14 (middle row). Kinesthetic teach-in means "taking the robot by the hand", performing the task by moving the robot while it is in gravity-compensation mode and recording the joint angles, velocities and accelerations. From the imitation, it can be determined by cross-validation that 31 shape-parameters per motor primitive are needed. As expected, the robot fails to reproduce the presented behavior and reinforcement learning is needed for self-improvement.

Figure 4.13 shows the expected return over the number of rollouts where convergence to a maximum is clearly recognizable. The robot regularly succeeds at bringing the ball into the cup after approximately 75 rollouts. Figure 4.15 shows the improvement of the policy over the rollouts. A nine year old child got the ball in the cup for the first time after 35 trials while the robot got the ball in for the first time after 42 rollouts.

Figure 4.14: This figure shows schematic drawings of the Ball-in-a-Cup motion, a kinesthetic teach-in as well as the final learned robot motion. The green arrows show the directions of the current movements in that frame. The human cup motion was taught to the robot by imitation learning with 31 parameters per joint for an approximately three seconds long trajectory. The robot manages to reproduce the imitated motion quite accurately, but the ball misses the cup by several centimeters. After approximately 42 rollouts of our Policy learning by Weighting Exploration with the Returns (PoWER) algorithm the robot has improved its motion to so that the ball goes into the cup. After approximately 75 rollouts we have good performance and at the end of the 100 rollouts we have virtually no failures anymore. Also see Figure 4.13 and 4.15.

(a) Initial policy after imitation learning.
(b) Policy after 15 rollouts, already closer.
(c) Policy after 25 rollouts, going too far.
(d) Policy after 45 rollouts, hitting the near rim.
(e) Policy after 60 rollouts, hitting the far rim.
(f) Final policy after 100 rollouts.

Figure 4.15: This figure shows the improvement of the policy over rollouts. The snapshots from the video show the position of the ball closest to the cup during a rollout.

4.5 Learning Perceptual Coupling

We have applied the presented algorithm in order to teach a physically realistic simulation of an anthropomorphic SARCOS robot arm how to perform Ball-in-a-Cup (for a description see Section 4.4). Here we use a different motion, see Figure 4.16(top) for an illustrative figure. For this evaluation, the state of the system is described in the Cartesian coordinates of the ball as in Section 4.4 but we use the Cartesian coordinates of the cup instead of the robot joint angles (i.e., the operational space instead of the joint space). The actions are the cup accelerations in Cartesian coordinates with each direction represented by a motor primitive (instead of the accelerations and motor primitives for the joints). An operational space control law [Nakanishi et al., 2007] is used in order to transform accelerations in the operational space of the cup into joint space torques. All motor primitives are perturbed separately but employ the same joint reward which is exactly the same as in Section 4.4 except the scaling parameter which we set to $\alpha = 10000$ for this evaluation as we have to cope with rollouts that are far off. As mentioned in Section 4.4, a small perturbation of the initial condition or the trajectory will drastically change the movement of the ball and hence the outcome of the rollout if we do not use any form of perceptual coupling to the external variable "ball".

We recorded the motions of a human player using a VICON$^{\text{TM}}$ motion-capture setup in order to obtain an example for imitation as shown in Figure 4.16(bottom). We used one of the recorded trajectories for which, when played back in simulation, the ball goes in but does not pass the center of the opening of the cup and, thus, does not optimize the reward. This movement is then used for initializing the motor primitives and determining their parametric structure where cross-validation indicates that 91 shape-parameters per motor primitive are optimal from a bias-variance point of view. The trajectories are optimized by reinforcement learning using the PoWER algorithm on the parameters $\mathbf{w}$ for non perturbed initial conditions as in Section 4.4. Also for this motion the robot constantly succeeds at bringing the ball into the cup after approximately 60-80 rollouts given no noise and perfect initial conditions in simulation (for this part).

One set of the found trajectories is then used to calculate the mean motion $\bar{\mathbf{y}} = (\mathbf{h} - \mathbf{b})$ and $\dot{\bar{\mathbf{y}}} = (\dot{\mathbf{h}} - \dot{\mathbf{b}})$, where $\mathbf{h}$ and $\mathbf{b}$ are the hand and ball trajectories. These correspond to the positions and velocities of the ball relative to the cup. This set of trajectories is also used as the standard cup trajectories. For perfect initial conditions and no noise the robot will perform the learned trajectory, as $\bar{\mathbf{y}} = 0$ and $\dot{\bar{\mathbf{y}}} = 0$ the perceptual coupling will not influence the trajectory. If there are any deviations from this defined trajectory, the perceptual coupling will move the cup in order to regain the desired relative positions and velocities. For example, if the ball is a bit to the right of the desired relative position, the perceptual coupling could compensate by moving the cup a bit to the right.

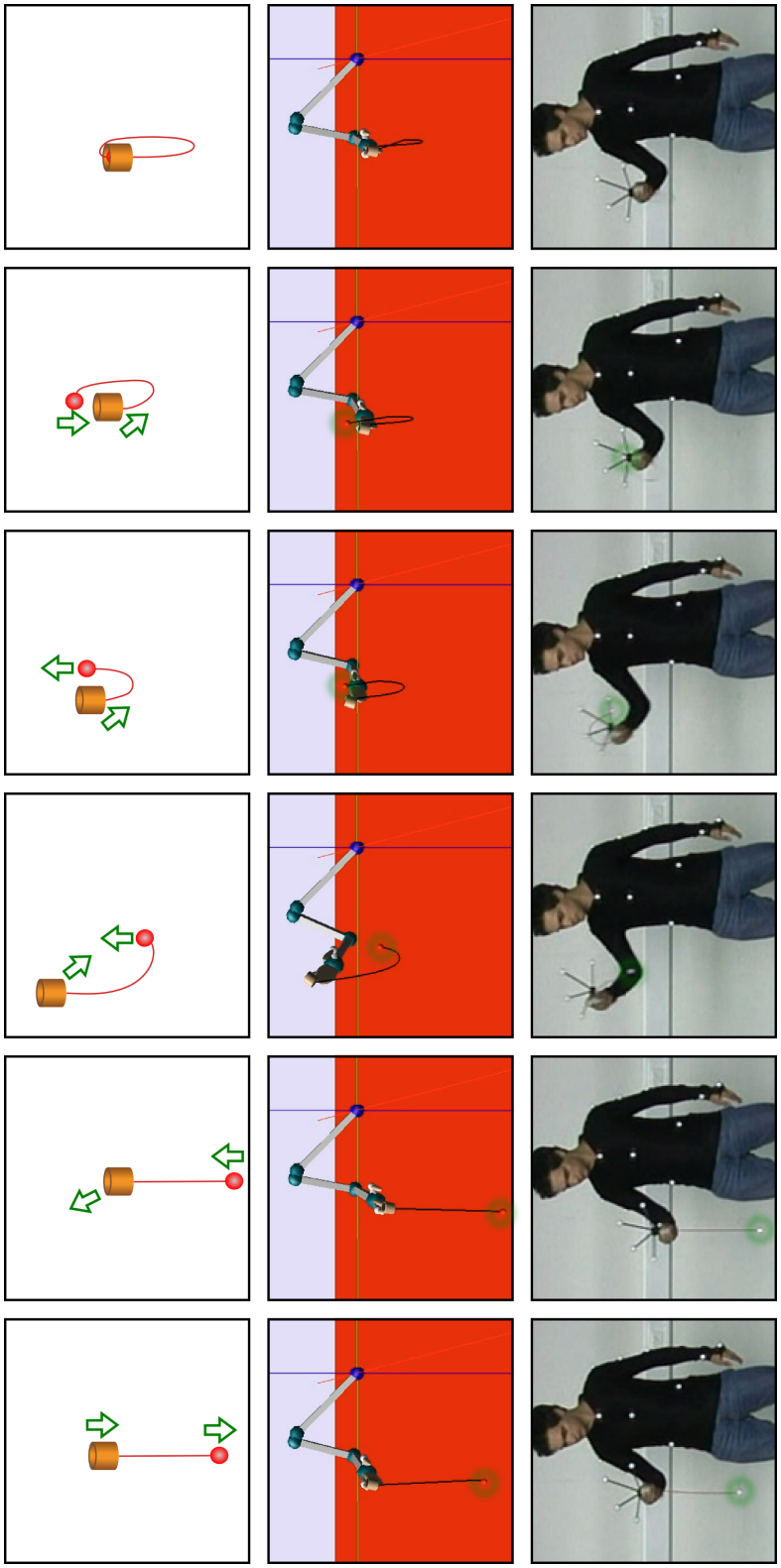


Figure 4.16: This figure shows schematic drawings of the Ball-in-a-Cup motion, the final learned robot motion as well as a motion-captured human motion. The green arrows show the directions of the current movements in that frame. The human cup motion was taught to the robot by imitation learning with 91 parameters for an approximately 1.5 seconds long trajectory.

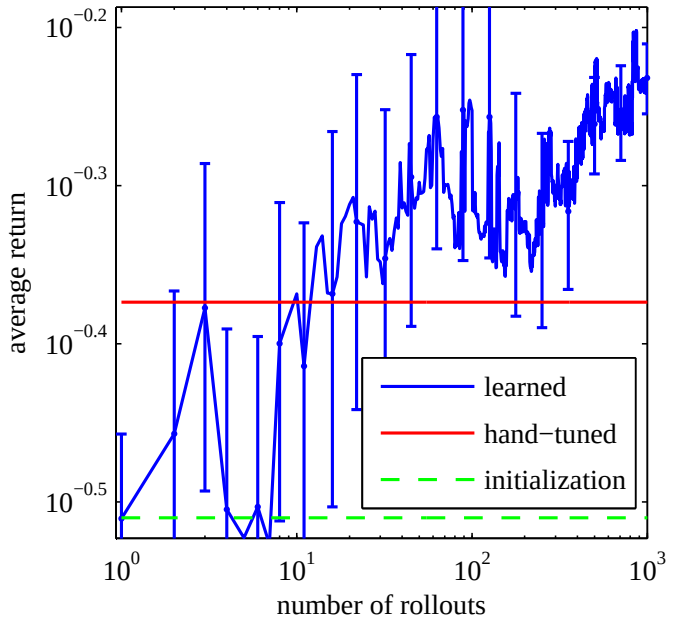


Figure 4.17: This figure shows the expected return of the learned policy in the Ball-in-a-Cup scenario (averaged over 3 runs with different random seeds and the standard deviation indicated by the error bars) for a specific test scenario. The convergence is not uniform as the algorithm is optimizing the returns for a whole range of perturbations and not just for the test case. Thus, the variance in the return as the improved policy might get worse for the test case but improve over all cases. Our algorithm rapidly improves, regularly beating a hand-tuned solution after less than fifty rollouts and converging after approximately 600 rollouts. Note that this plot is a double logarithmic plot and, thus, single unit changes are significant as they correspond to orders of magnitude.

Hand tuned coupling factors can cope with small perturbations of the initial conditions. In order to make the coupling more robust, we use reinforcement learning with the same rewards as before. The initial conditions (i.e., both positions and velocities) of the ball are perturbed completely randomly using Gaussian random values with variances set according to the desired stability region. We did not use the PEGASUS Trick of reinitializing the random numbers and, thus, training on a defined set of initial conditions. For the training we perturbed the initial conditions of the ball with Gaussian-distributed noise of concurrent standard deviations of 0.01m for x and y and of 0.1 m/s for $\dot{x}$ and $\dot{y}$. The PoWER algorithm converges after approximately 600-800 rollouts for 13 perceptual coupling parameters per degree of freedom (Figure 4.17). This is equivalent to roughly 45 minutes of continuous training for a human player. After a training session of this duration a ten year old child is usually able to get the ball in the cup almost every time, even with

small perturbations. The learned perceptual coupling manages to perform successful rollouts for all tested cases where the hand-tuned coupling was also successful. The learned coupling pushes the limits of the canceled perturbations significantly further and still performs consistently well for double the standard deviations seen in the reinforcement learning process. Figure 4.18 shows an example of how the visual coupling adapts the hand trajectories in order to cancel perturbations and to get the ball in the cup.

We also learned the coupling directly in joint-space in order to show, that the augmented motor primitives can handle perception and action in different spaces (perception in task space and action in joint space, for our evaluation). For each of the seven degrees of freedom a separate motor primitive is used, $\bar{\mathbf{y}}$ and $\dot{\bar{\mathbf{y}}}$ remain the same as before. Here we were not able to find good coupling factors by hand-tuning. Reinforcement learning finds working parameters but they do not perform as well as the Cartesian version. These effects can be explained by two factors: the learning task is harder as we have a higher dimensionality. Furthermore, we are learning the inverse kinematics of the robot implicitly. If the perturbations are large, the perceptual coupling has to do large corrections. These large corrections tend to move the robot in regions where the inverse kinematics differ from the ones for the mean motion and, thus, the learned implicit inverse kinematics no longer perform well. This behavior leads to even larger deviations and the effects accumulate.

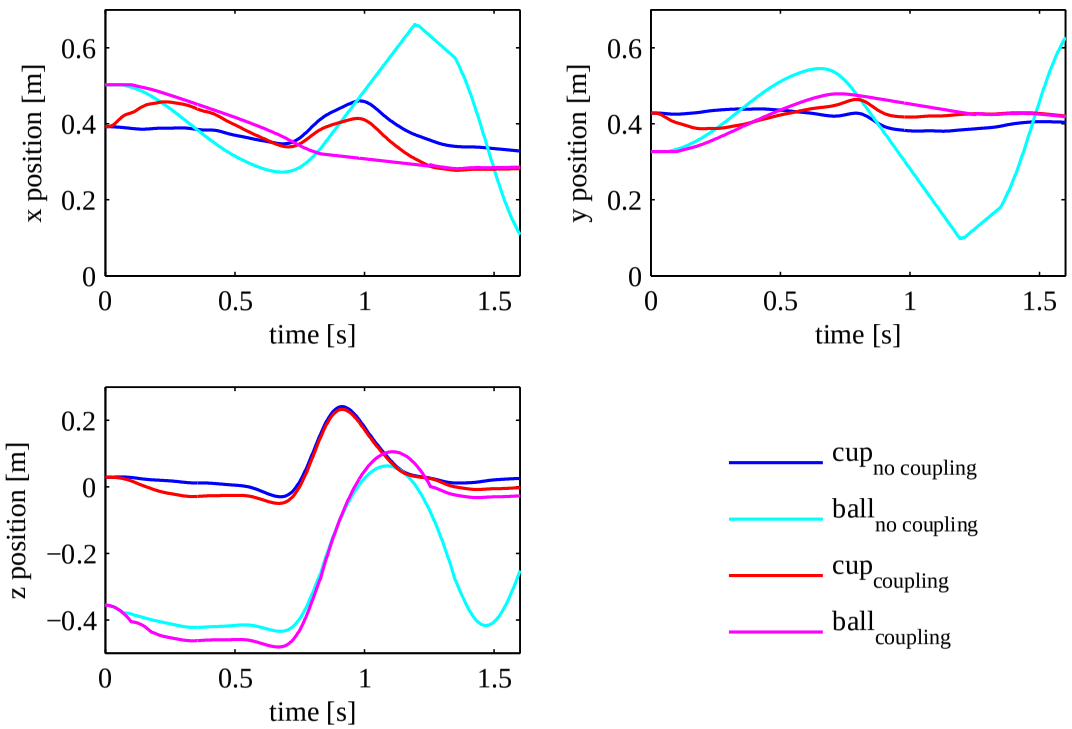


Figure 4.18: This figure compares cup and ball trajectories with and without perceptual coupling. The trajectories and different initial conditions are clearly distinguishable. The perceptual coupling cancels the swinging motion of the string and ball "pendulum" out. The successful rollout is marked either by overlying (x and y) or parallel (z) trajectories of the ball and cup from 1.2 seconds on.

5 Conclusion

In this diploma thesis, we have presented a new perspective on policy learning methods and an application to a highly complex motor learning task on a real Barrett WAM^{TM} robot arm. We have generalized the previous work in [Dayan and Hinton, 1997, Peters and Schaal, 2007b] from the immediate reward case to the episodic case. In the process, we could show that policy gradient methods are a special case of this more general framework. During initial experiments, we realized that state-dependent exploration allows the derivation of much faster policy learning methods than the common Gaussian exploration with constant variance. This resulted in a novel policy learning algorithm, Policy learning by Weighting Exploration with the Returns (PoWER), an EM-inspired algorithm that outperforms several other policy search methods on standard benchmarks as well as on a simulated Underactuated Swing-Up.

We successfully applied this novel PoWER algorithm in the context of learning two tasks on a physical robot, i.e., the Underacted Swing-Up and Ball-in-a-Cup. Due to the "curse of dimensionality", we cannot start with an arbitrary solution as we would have to try infinitely many solutions. Instead, we mimic the way children learn Ball-in-a-Cup and first present an example for imitation learning which is recorded using kinesthetic teach-in or motion-capture. Subsequently, our reinforcement learning algorithm takes over and learns how to move the ball into the cup reliably. After a number of rollouts comparable to the learning rate of a ten year old child, the task can be regularly fulfilled and the robot shows very good average performance.

Perceptual coupling for motor primitives is an important topic as it results in more general and more reliable solutions while it allows the application of the framework of motor primitive based on dynamical systems to many other motor control problems. As manual tuning can only work in limited setups, an automatic acquisition of this perceptual coupling is essential.

Therefore, we have contributed an augmented version of the motor primitive framework originally suggested by Ijspeert et al. [2002a, 2003], Schaal et al. [2007] in this diploma thesis such that it incorporates perceptual coupling while keeping a distinctively similar structure to the original approach and, thus, preserving most of the important properties. The resulting framework works well for learning Ball-in-a-Cup on a simulated anthropomorphic SARCOS arm in setups where the original motor primitive framework would not suffice to fulfill the task.

5.1 Future Work

Motor primitives are elemental motions which are described in detail in Section 3. To date these are mostly used for repetitive tasks. The current formulation of motor primitives will have to be extended to include a supervisory layer for selection, sequencing and superposition of motor primitives for accomplishing multiple tasks. Basic sequencing with linear blending as well as independently superposed motor primitives are straightforward to implement but in several cases these approaches will not suffice. For a versatile framework, sequencing and blending should take into account physical limitations (e.g., impossible movements) and as well allow optimization according to defined cost functions. For superposition considering the possibility of mutual influence between several primitives is necessary. The methods introduced in this diploma thesis will serve to learn the motor primitives for a "motion library" from which the supervisory layer draws the motor primitives.

The motor primitives also have to be adapted according to external triggers. Here the focus will lie on the state of the object we want to interact with. Two cases have to be distinguished: only one single point of the trajectory matters (i.e., the end point or a via-point) or the whole trajectory has to be adapted. This diploma theses introduced an augmented version of the motor primitives for the second case. The presented novel policy learning algorithm PoWER can also most likely be applied to learn both the mean motor primitives and the mapping of the amplitude and goal parameters to the desired end point or via-point.

The simplest and fastest learning method is imitation, either observed directly from a human performer or taught by a kinesthetic teach-in. Depending on the task a single example is sufficient or several are needed. We want to learn perceptual coupling directly by observation for simple examples. In order to further improve performance (this can range from the robot not succeeding in repeating the task at all to performing a task more efficiently) reinforcement learning will be needed. Policy gradient and EM like policy learning methods have proven promising, as shown in this diploma thesis, and shall be examined, extended and developed.

Bibliography

P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-First International Conference on Machine Learning (ICML)*, New York, NY, USA, 2004. ACM. ISBN 1-58113-828-5. URL http://ai.stanford.edu/~ang/papers/icml04-apprentice.ps.

C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50(1):5–43, 2003. URL http://www.cs.ubc.ca/~nando/papers/mlintro.ps.

H. Arisumi, K. Yokoi, and K. Komoriya. Kendama game by casting manipulator. In *Proceedings of the IEEE/RSJ 2005 International Conference on Intelligent RObots and Systems (IROS)*, 2005. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1544972.

C. G. Atkeson. Using local trajectory optimizers to speed up global optimization in dynamic programming. In J. E. Hanson, S. J. Moody, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 6 (NIPS)*, pages 503–521, Denver, CO, USA, 1994. Morgan Kaufmann. URL ftp://ftp.cc.gatech.edu/pub/people/cga/local.ps.gz.

H. Attias. Planning by probabilistic inference. In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics (AISTATS)*, Key West, FL, USA, 2003. URL http://research.microsoft.com/conferences/aistats2003/proceedings/206.pdf.

J. Bagnell and J. Schneider. Covariant policy search. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1019–1024, Acapulco, Mexico, August 2003. URL http://dli.iiit.ac.in/ijcai/IJCAI-2003/PDF/146.pdf.

J. Bagnell, S. Kadade, A. Ng, and J. Schneider. Policy search by dynamic programming. In S. Thrun, L. K. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16 (NIPS)*, Vancouver, BC, CA, 2003. Cambridge, MA: MIT Press. URL http://ai.stanford.edu/~ang/papers/nips03-dpps.ps.

J. Baxter and P. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001. URL http://www.jair.org/media/806/live-806-1942-jair.pdf.

J. Baxter, P. Bartlett, and L. Weaver. Experiments with infinite-horizon, policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:351–381, 2001. URL http://www.jair.org/media/807/live-807-1945-jair.pdf.

R. E. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957. URL http://portal.acm.org/citation.cfm?id=862270.

S. Ben-Itzhak and A. Karniel. Minimum acceleration criterion with constraints implies bang-bang control as an underlying principle for optimal trajectories of arm reaching movements. *Neural Computation*, 20(3):779–812, March 2008. URL http://neco.mitpress.org/cgi/reprint/20/3/779.pdf.

J. Binder, D. Koller, S. Russell, and K. Kanazawa. Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29(2–3):213–244, 1997. URL http://ai.stanford.edu/~koller/Papers/Binder+al:MLJ97.pdf.

E. Bizzi. Motor primitives and rehabilitation. In *Proceedings of the 6th International Workshop on Virtual Rehabilitation (IWVR)*, pages 20–22, 2007. URL http://ieeexplore.ieee.org/iel5/4362118/4362119/04362123.pdf.

E. Bizzi, A. d'Avella, P. Saltiel, and M. C. Tresch. Modular organization of spinal motors systems. *The Neuroscientist*, 8(5):437–442, 2002. URL http://web.mit.edu/bcs/bizzilab/publications/bizzi2002.pdf.

S. Calinon, F. Guenter, and A. Billard. On learning, representing and generalizing a task in a humanoid robot. *IEEE Transactions on Systems, Man and Cybernetics, Part B. Special issue on robot learning by observation, demonstration and imitation*, 37(2):286–298, 2007. URL http://infoscience.epfl.ch/record/102365/files/Calinon-JSMC2006.pdf.

P. Dayan and G. E. Hinton. Using expectation-maximization for reinforcement learning. *Neural Computation*, 9(2):271–278, 1997. URL citeseer.ist.psu.edu/dayan97using.html.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B (Methodological)*, 39:1–38, 1977. URL http://links.jstor.org/sici?sici=0035-9246%281977%2939%3A1%3C1%3AMLFIDV%3E2.0.CO%3B2-Z.

E. Drumwright and M. J. Matarić. Generating and recognizing freespace movements in humanoid robots. In *Proceedings of the IEEE/RSJ 2003 International Conference on Intelligent RObots and Systems (IROS)*, pages 1672–1678, 2003. URL citeseer.ist.psu.edu/drumwright03generating.html.

E. Drumwright, O. C. Jenkins, and M. J. Matarić. Exemplar-based primitives for humanoid movement classification and control. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 140–145. IEEE, 2004. URL http://cres.usc.edu/pubdb_html/files_upload/356.pdf.

A. El-Fakdi, M. Carreras, and P. Ridao. Towards direct policy search reinforcement learning for robot control. In *Proceedings of the IEEE/RSJ 2006 International Conference on Intelligent RObots and Systems (IROS)*, Beijing, China, October 9-15 2006. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4058885.

T. Flash and B. Hochner. Motor primitives in vertebrates and invertebrates. *Current Opinions in Neurobiology*, 15:660–666, 2005. URL http://linkinghub.elsevier.com/retrieve/pii/S0959438805001601.

S. M. Fogel and C. T. Smith. Learning-dependent changes in sleep spindles and stage 2 sleep. *Journal of Sleep Research*, 15(3):250–255, 2006. doi: 10.1111/j.1365-2869.2006.00522.x. URL http://www3.interscience.wiley.com/journal/118589642/abstract.

T. Fujii, T. Yasuda, S. Yokoi, and J.-i. Toriwaki. A virtual pendulum manipulation system on a graphic workstation. In *Proceedings of the 2nd IEEE International Workshop on Robot and Human Communication (RO-MAN)*, 1993. URL http://ieeexplore.ieee.org/iel2/2970/8413/00367704.pdf.

Z. Ghahramani and M. I. Jordan. Supervised learning from incomplete data via an EM approach. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6 (NIPS)*, pages 120–127, San Mateo, CA, 1994. Morgan Kaufmann. URL http://learning.eng.cam.ac.uk/zoubin/papers/nips93.ps.

N. J. Giordano and H. Nakanishi. *Computational Physics, 2nd Edition*. Prentice Hall, 2005. URL http://www.physics.purdue.edu/~hisao/book/.

S. F. Giszter, F. A. Mussa-Ivaldi, and E. Bizzi. Convergent force fields organized in the frog's spinal cord. *Journal of Neuroscience*, 13(2):467, 1993. URL http://web.mit.edu/bcs/bizzilab/publications/giszter1993.pdf.

S. F. Giszter, K. A. Moxon, I. A. Rybak, and J. K. Chapin. A neurobiological perspective on humanoid robot design. *IEEE Intelligent Systems*, 15(4):64–69, 2000. URL http://neurobio.drexelmed.edu/Rybakweb/ieee_is.pdf.

F. Guenter, M. Hersch, S. Calinon, and A. Billard. Reinforcement learning for imitating constrained reaching movements. *Advanced Robotics, Special Issue on Imitative Robots*, 21(13):1521–1544, 2007. URL http://infoscience.epfl.ch/record/114046/files/Guenter_AR07.pdf.

V. Gullapalli, J. Franklin, and H. Benbrahim. Aquiring robot skills via reinforcement learning. *IEEE Control Systems Journal, Special Issue on Robotics: Capturing Natural Motion*, 4(1):13–24, February 1994. URL http://ieeexplore.ieee.org/iel1/37/6540/00257890.pdf.

V. Hamburger, K. Berns, F. Iida, and R. Pfeifer. Standing up with motor primitives. In *Proceedings of the International Conference on Climbing and Walking Robots*

(CLAWAR), pages 383–390, 2005. URL http://people.csail.mit.edu/iida/papers/hamburger_clawar05.pdf.

M. Hoffman, A. Doucet, N. de Freitas, and A. Jasra. Bayesian policy learning with trans-dimensional mcmc. In *Advances in Neural Information Processing Systems 20 (NIPS)*, Vancouver, BC, CA, 2007. URL http://www.cs.ubc.ca/~nando/papers/pomdp1.pdf.

A. J. Ijspeert, J. Nakanishi, and S. Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 1398–1403, Washington, DC, May 11-15 2002a. URL http://www-clmc.usc.edu/publications/I/ijspeert-ICRA2002.pdf.

A. J. Ijspeert, J. Nakanishi, and S. Schaal. Learning rhythmic movements by demonstration using nonlinear oscillators. In *Proceedings of the IEEE/RSJ 2002 International Conference on Intelligent RObots and Systems (IROS)*, pages 958–963, Lausanne, Sept.30-Oct.4 2002b. Piscataway, NJ: IEEE. URL http://www-clmc.usc.edu/publications/I/ijspeert-IROS2002.pdf.

A. J. Ijspeert, J. Nakanishi, and S. Schaal. Learning attractor landscapes for learning motor primitives. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 16 (NIPS)*, volume 15, pages 1547–1554, Cambridge, MA, 2003. MIT Press. URL http://www-clmc.usc.edu/publications/I/ijspeert-NIPS2002.pdf.

T. Inamura, I. Toshima, H. Tanie, and Y. Nakamura. Embodied symbol emergence based on mimesis theory. *International Journal of Robotics Research*, 23(4-5):363–377, April–May 2004. URL http://www.ynl.t.u-tokyo.ac.jp/publications/pdf2004/inamura-ijrr.pdf.

M. Kawato, F. Gandolfo, H. Gomi, and Y. Wada. Teaching by showing in kendama based on optimization principle. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, volume 1, pages 601–606, Sorrento, Italy, 1994. URL ieeexplore.ieee.org/iel3/3988/11478/00531981.pdf.

D. E. Kirk. *Optimal control theory*. Prentice-Hall, Englewood Cliffs, New Jersey, 1970. URL http://store.doverpublications.com/0486434842.html.

J. Kober and J. Peters. Reinforcement learning of perceptual coupling for motor primitives. In *8th European Workshop on Reinforcement Learning (EWRL)*, pages 1–8, 07 2008a. URL http://ewrl08.futurs.inria.fr/.

J. Kober and J. Peters. Policy search for motor primitives in robotics. In *Advances in Neural Information Processing Systems (NIPS)*, 2008b. URL http://nips.cc/Conferences/2008/.

J. Kober, B. Mohler, and J. Peters. Learning perceptual coupling for motor primitives. In *Proceedings of the IEEE/RSJ 2008 International Conference on Intelligent*

RObots and Systems (IROS), pages 834–839, Nice, France, 2008. URL http://www.kyb.mpg.de/publications/attachments/IROS2008-Kober_[0].pdf.

N. Kohl and P. Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2619–2624, New Orleans, LA, May 2004. URL http://www.cs.utexas.edu/~pstone/Papers/bib2html-links/icra04.pdf.

J. Z. Kolter, A. Coates, A. Y. Ng, Y. Gu, and C. DuHadway. Space-indexed dynamic programming: Learning to follow trajectories. In *Proceedings of the Twenty-Fifth International Conference on Machine Learning (ICML)*, 2008. URL http://www.stanford.edu/~kolter/pubs/kolter-icml08.pdf.

J. Konczak. On the notion of motor primitives in humans and robots. *Lund University Cognitive Studies*, 123:47–53, 2005. URL http://www.lucs.lu.se/LUCS/123/Konczak.pdf.

I. Kwee, M. Hutter, and J. Schmidhuber. Gradient-based reinforcement planning in policy-search methods. In M. A. Wiering, editor, *Proceedings of the 5th European Workshop on Reinforcement Learning (EWRL)*, number 27 in Cognitieve Kunstmatige Intelligentie, pages 27–29, Manno(Lugano), CH, 2001. Onderwijsinsituut CKI - Utrecht University. ISBN 90-393-2874-9. URL citeseer.ist.psu.edu/article/kwee01gradientbased.html.

G. Lawrence, N. Cowan, and S. Russell. Efficient gradient estimation for motor control learning. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 354–361, Acapulco, Mexico, 2003. URL http://www.cs.berkeley.edu/~russell/papers/uai03-motor.pdf.

M. A. Lemay and W. M. Grill. Endpoint force patterns evoked by intraspinal stimulation- ipsilateral and contralateral responses in the cat. In *Proceedings of the World Congress on Medical Physics and Biomedical Engineering (WC)*, 2000. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=897870.

J. Marcoux and S. Rossignol. Initiating or blocking locomotion in spinal cats by applying noradrenergic drugs to restricted lumbar spinal segments. *The Journal of Neuroscience*, 20(22):8577–8585, 2000. URL http://www.jneurosci.org/cgi/reprint/20/22/8577.pdf.

M. J. Matarić. Sensory-motor primitives as a basis for imitation: Linking perception to action and biology to robotics. *Imitation in Animals and Artifacts*, pages 391–422, 2002. URL http://robotics.usc.edu/~maja/publications/imit.ps.gz.

G. J. McLachan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley Series in Probability and Statistics. John Wiley & Sons, 1997. URL http://eu.wiley.com/WileyCDA/WileyTitle/productCd-0471201707.html.

C. E. Milner, S. M. Fogel, and K. A. Cote. Habitual napping moderates motor performance improvements following a short daytime nap. *Biological Psy-*

chology, 73(2):141–156, 2006. doi: 10.1016/j.biopsycho.2006.01.015. URL http://linkinghub.elsevier.com/retrieve/pii/S0301051106000299.

H. Miyamoto, S. Schaal, F. Gandolfo, H. Gomi, Y. Koike, R. Osu, E. Nakano, Y. Wada, and M. Kawato. A kendama learning robot based on bi-directional theory. *Neural Networks*, 9(8):1281–1302, 1996. ISSN 0893-6080. doi: http://dx.doi.org/10.1016/S0893-6080(96)00043-3. URL http://www-clmc.usc.edu/publications/M/miyamoto-NN1996.pdf.

F. A. Mussa-Ivaldi, S. F. Giszter, and E. Bizzi. Linear combinations of primitives in vertebrate motor control. *Proceedings of the National Academy of Sciences of the United States of America*, 91:7534–7538, 1994. URL http://web.mit.edu/bcs/bizzilab/publications/mussa-ivaldi1994.pdf.

J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato. A framework for learning biped locomotion with dynamic movement primitives. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS)*, Los Angeles, CA: Nov.10-12, Santa Monica, CA, 2004a. IEEE. URL http://www-clmc.usc.edu/publications/N/nakanishi-ICHR2004.pdf.

J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato. Learning from demonstration and adaptation of biped locomotion. *Robotics and Autonomous Systems (RAS)*, 47(2-3):79–91, 2004b. URL http://www-clmc.usc.edu/publications/N/nakanishi-RAS2004.pdf.

J. Nakanishi, M. Mistry, J. Peters, and S. Schaal. Experimental evaluation of task space position/orientation control towards compliant control for humanoid robots. In *Proceedings of the IEEE/RSJ 2007 International Conference on Intelligent RObotics Systems (IROS)*, 2007. URL http://www-clmc.usc.edu/publications/T/nakanishi-IROS2007.pdf.

A. Y. Ng and M. Jordan. Pegasus: A policy search method for large mdps and pomdps. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 406–415, Palo Alto, CA, 2000. URL http://ai.stanford.edu/~ang/papers/uai00-pegasus.pdf.

L. Peshkin. *Reinforcement Learning by Policy Search*. PhD thesis, Brown University, Providence, RI, 2001. URL citeseer.ist.psu.edu/article/peshkin00reinforcement.html.

J. Peters. *Machine Learning of Motor Skills for Robotics*. PhD thesis, University of Southern California, Los Angeles, CA, 90089, USA, April 2007. URL http://eprints.pascal-network.org/archive/00003593/.

J. Peters and S. Schaal. Reinforcement learning for parameterized motor primitives. In *Proceedings of the 2006 International Joint Conference on Neural Networks (IJCNN)*, 2006a. URL http://www-clmc.usc.edu/publications/P/peters-IJCNN2006.pdf.

J. Peters and S. Schaal. Learning to control in operational space. *International Journal of Robotics Research*, 27(2):197–212, 02 2008a. URL http://ijr.sagepub.com/cgi/reprint/27/2/197.

J. Peters and S. Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 03 2008b. URL http://www-clmc.usc.edu/publications/P/peters-NC2008.pdf.

J. Peters and S. Schaal. Policy gradient methods for robotics. In *Proceedings of the IEEE/RSJ 2006 International Conference on Intelligent RObots and Systems (IROS)*, pages 2219 – 2225, Beijing, China, 2006b. URL http://www-clmc.usc.edu/publications/P/peters-IROS2006.pdf.

J. Peters and S. Schaal. Learning operational space control. In W. Burgard, G. S. Sukhatme, and S. Schaal, editors, *Proceedings of Robotics: Science and Systems (R:SS)*. Cambridge, MA: MIT Press, 2006c. URL http://www-clmc.usc.edu/publications/P/peters-RSS2006.pdf.

J. Peters and S. Schaal. Using reward-weighted regression for reinforcement learning of task space control. In *Proceedings of the IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL)*, Honolulu, HI, 2007a. URL http://www-clmc.usc.edu/publications/P/peters-ADPRL2007.pdf.

J. Peters and S. Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the International Conference on Machine Learning (ICML)*, Corvallis, OR, USA, 2007b. URL http://www-clmc.usc.edu/publications/P/peters_ICML2007.pdf.

J. Peters and S. Schaal. Reinforcement learning for operational space. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, Rome, Italy, 2007c. URL http://www-clmc.usc.edu/publications/P/peters-ICRA2007.pdf.

J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 05 2008c. URL http://www-clmc.usc.edu/publications/P/peters-NN2008.pdf.

J. Peters and S. Schaal. Applying the episodic natural actor-critic architecture to motor primitive learning. In *Proceedings of the 2007 European Symposium on Artificial Neural Networks (ESANN)*, 2007d. URL http://www-clmc.usc.edu/publications/P/peters-ESANN2007.pdf.

J. Peters and S. Schaal. Policy learning for motor skills. In *Proceedings of the 14th International Conference on Neural Information Processing (ICONIP)*, 2007e. URL http://www-clmc.usc.edu/publications/P/peters-ICONIP2007.pdf.

J. Peters, S. Vijayakumar, and S. Schaal. Reinforcement learning for humanoid robotics. In *Proceedings of the IEEE-RAS International Conference on Humanoid*

Robots (HUMANOIDS), pages 103–123, Karlsruhe, Germany, September 2003a. URL http://www-clmc.usc.edu/publications/p/peters-ICHR2003.pdf.

J. Peters, S. Vijayakumar, and S. Schaal. Scaling reinforcement learning paradigms for motor learning. In *Proceedings of the 10th Joint Symposium on Neural Computation (JSNC)*, Irvine, CA, May 2003b. URL http://www-clmc.usc.edu/publicatons/P/peters-JSNC2003.pdf.

J. Peters, S. Vijayakumar, and S. Schaal. Natural actor-critic. In *Proceedings of the European Conference on Machine Learning (ECML)*, pages 280–291, Porto, Portugal, 2005. URL http://www-clmc.usc.edu/publications/P/peters-ECML2005.pdf.

J. Peters, E. Theodorou, and S. Schaal. Policy gradient methods for machine learning. In *Proceedings of the INFORMS Conference of the Applied Probability Society (ICAPS)*, 2007. URL http://appliedprob.society.informs.org/INFORMS2007/proceedingsInforms2007.pdf.

J. Peters, J. Kober, and D. Nguyen-Tuong. Policy learning: a unified perspective with applications in robotics. In *Proceedings of the 8th European Workshop on Reinforcement Learning (EWRL)*, pages 1–8, 07 2008. URL http://ewrl08.futurs.inria.fr/.

D. Pongas, A. Billard, and S. Schaal. Rapid synchronization and accurate phase-locking of rhythmic motor primitives. In *Proceedings of the IEEE 2005 International Conference on Intelligent RObots and Systems (IROS)*, volume 2005, pages 2911–2916, 2005. URL http://ieeexplore.ieee.org/iel5/10375/32977/01545257.pdf.

M. Riedmiller, J. Peters, and S. Schaal. Evaluation of policy gradient methods and variants on the cart-pole benchmark. In *Proceedings of the 2007 IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL)*, 2007. URL http://www-clmc.usc.edu/publications/P/riedmiller-ADPRL2007.pdf.

L. Righetti and A. J. Ijspeert. Programmable central pattern generators: an application to biped locomotion control. In *Proceedings of the 2006 IEEE International Conference on Robotics and Automation (ICRA)*, 2006. URL http://ieeexplore.ieee.org/iel5/10932/34383/01641933.pdf.

C. Rose, M. Cohen, and B. Bodenheimer. Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics and Applications*, 18, no.5: 32–40, 1998. URL http://graphics.cs.cmu.edu/nsp/course/15-464/Fall05/papers/rose98verbs.pdf.

T. Sakaguchi and F. Miyazaki. Dynamic manipulation of ball-in-cup game. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 2941–2948, 1994. URL ftp://ftp.cc.gatech.edu/pub/gvu/tr/2000/00-23.pdf.

T. D. Sanger. Optimal movement primitives. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1023–1030. MIT Press, 1994. URL http://comptop.stanford.edu/references/s.pdf.

S. Sato, T. Sakaguchi, Y. Masutani, and F. Miyazaki. Mastering of a task with interaction between a robot and its environment : "kendama" task. *Transactions of the Japan Society of Mechanical Engineers. C*, 59(558):487–493, 1993. ISSN 03875024. URL http://ci.nii.ac.jp/naid/110002380882/en/.

S. Schaal. Learning from demonstration. In M. Mozer, M. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 9, pages 1040–1046, Cambridge, MA, 1997. MIT Press. URL http://www-clmc.usc.edu/publications/S/schaal-NIPS1997.pdf.

S. Schaal. The SL simulation and real-time control software package. Technical report, University of Southern California, 2004. URL http://www-clmc.usc.edu/publications/S/schaal-TRSL.pdf.

S. Schaal, J. Peters, J. Nakanishi, and A. J. Ijspeert. Control, planning, learning, and imitation with dynamic movement primitives. In *Proceedings of the Workshop on Bilateral Paradigms on Humans and Humanoids, IEEE 2003 International Conference on Intelligent RObots and Systems (IROS)*, Las Vegas, NV, Oct. 27-31, 2003. URL http://www-clmc.usc.edu/publications/S/schaal-IROS2003.pdf.

S. Schaal, J. Peters, J. Nakanishi, and A. J. Ijspeert. Learning movement primitives. In *International Symposium on Robotics Research 2003 (ISRR)*, Springer Tracts in Advanced Robotics, pages 561–572, Ciena, Italy, 2004. Springer. URL http://www-clmc.usc.edu/publications/S/schaal-ISRR2003.pdf.

S. Schaal, P. Mohajerian, and A. J. Ijspeert. Dynamics systems vs. optimal control — a unifying view. *Progress in Brain Research*, 165(1):425–445, 2007. URL http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?cmd=Retrieve&db=PubMed&dopt=Citation&list_uids=17925262.

T. Shone, G. Krudysz, and K. Brown. Dynamic manipulation of kendama. Technical report, Rensselaer Polytechnic Institute, 2000. URL http://www.oleblue.com/kendama.htm.

M. Strens and A. Moore. Direct policy search using paired statistical tests. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, 2001. URL http://www.autonlab.org/papers/strens-2001.pdf.

C. Sumners. *Toys in Space: Exploring Science with the Astronauts*. McGraw-Hill, 1997. URL http://eric.ed.gov/ERICWebPortal/recordDetail?accno=ED390652.

R. Sutton and A. Barto. *Reinforcement Learning*. MIT PRESS, 1998. URL http://www.cs.ualberta.ca/~sutton/book/the-book.html.

R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In S. A. Solla, T. K. Leen, and K.-R. Mueller, editors, *Advances in Neural Information Processing Systems 13 (NIPS)*, pages 1057–1063, Denver, CO, 2000. MIT Press. URL http://www.cs.ualberta.ca/~sutton/papers/SMSM-NIPS99.pdf.

U. Syed and R. Schapire. A game-theoretic approach to apprenticeship learning. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20 (NIPS)*, pages 1449–1456, Cambridge, MA, 2008. MIT Press. URL http://books.nips.cc/papers/files/nips20/NIPS2007_0949.pdf.

K. Takenaka. Dynamical control of manipulator with vision : "cup and ball" game demonstrated by robot. *Transactions of the Japan Society of Mechanical Engineers. C*, 50(458):2046–2053, 1984. ISSN 03875024. URL http://ci.nii.ac.jp/naid/110002388116/.

M. E. Taylor, S. Whiteson, and P. Stone. Transfer via inter-task mappings in policy search reinforcement learning. In *Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, May 2007. URL http://www.cs.utexas.edu/~pstone/Papers/bib2html-links/AAMAS07-taylor.pdf.

R. Tedrake, T. W. Zhang, and H. S. Seung. Stochastic policy gradient reinforcement learning on a simple 3d biped. In *Proceedings of the IEEE 2004 International Conference on Intelligent RObots and Systems (IROS)*, pages 2849–2854, 2004. URL http://groups.csail.mit.edu/locomotion/publications/Tedrake04a.pdf.

K. A. Thoroughman and R. Shadmehr. Learning of action through adaptive combination of motor primitives. *Nature*, 407:742–747, 2000. URL http://www.nature.com/nature/journal/v407/n6805/pdf/407742a0.pdf.

E. Todorov and Z. Ghahramani. Unsupervised learning of sensory-motor primitives. In *Proceedings of the 25th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMB)*, 2003. URL http://www.cogsci.ucsd.edu/~todorov/papers/emb03b.pdf.

M. Toussaint and C. Goerick. Probabilistic inference for structured planning in robotics. In *Proceedings of the IEEE/RSJ 2007 International Conference on Intelligent RObots and Systems (IROS)*, San Diego, CA, USA, 2007. URL http://ml.cs.tu-berlin.de/~mtoussai/publications/toussaint-goerick-07-iros.pdf.

M. Toussaint, M. Gienger, and C. Goerick. Optimization of sequential attractor-based movement for compact behaviour generation. In *Proceedings of the 7th IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS)*, 2007. URL http://planning.cs.cmu.edu/humanoids07/p/42.pdf.

M. C. Tresch and E. Bizzi. Responses to microstimulation in the chronically spinalized rat and their relationship to spinal systems activated by low threshold cutaneous

systems. *Experimental Brain Research*, 129:401–416, 1999. URL http://web.mit.edu/bcs/bizzilab/publications/tresch1999a.pdf.

F. E. Udwadia and R. E. Kalaba. *Analytical Dynamics: A New Approach*. Cambridge University Press, Cambridge, UK, 1996. URL http://www.cambridge.org/us/catalogue/catalogue.asp?isbn=0521482178.

H. Urbanek, A. Albu-Schäffer, and P.v.d.Smagt. Learning from demonstration repetitive movements for autonomous service robotics. In *Proceedings of the IEEE/RSL 2004 International Conference on Intelligent RObots and Systems (IROS)*, Sendai, Japan, 2004. URL http://www.robotic.de/fileadmin/robotic/smagt/publications/UrbAlbSma2004.pdf.

L. Verlet. Computer "experiments" on classical fluids. i. thermodynamical properties of lennard-jones molecules. *Physical Review*, 159(1):98+, July 1967. doi: 10.1103/PhysRev.159.98. URL http://dx.doi.org/10.1103/PhysRev.159.98.

F. J. Vesely. *Computational Physics - An Introduction, Second Edition*. Kluwer Academic / Plenum Publishers, 2001. URL http://homepage.univie.ac.at/Franz.Vesely/cp_kluwer/index.html.

Wikipedia. Ball-in-a-cup, August 2008a. URL http://en.wikipedia.org/wiki/Ball_in_a_cup.

Wikipedia. Semi-implicit euler method, August 2008b. URL http://en.wikipedia.org/wiki/Semi-implicit_Euler_method.

B. Williams, M. Toussaint, and A. Storkey. Modelling motion primitives and their timing in biologically executed movements. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20 (NIPS)*, pages 1609–1616, Cambridge, MA, 2008. MIT Press. URL http://books.nips.cc/papers/files/nips20/NIPS2007_0665.pdf.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992. URL ftp://ftp.ccs.neu.edu/pub/people/rjw/conn-reinf-ml-92.ps.

G. Wulf. *Attention and motor skill learning*. Human Kinetics, Champaign, IL, 2007. URL http://www.humankinetics.com/products/showproduct.cfm?isbn=073606270x.

List of Algorithms

List of Figures

Abbreviations

In this diploma thesis we use the following mathematical notation throughout this thesis:

Notation	Description
$\{x_1, x_2, \ldots, x_n\}$	set with elements $x_1, x_2, \ldots, x_n$
$\mathbb{R}$	real numbers
$\mathbf{x} = [x_1, x_2, \ldots, x_n]$	a vector
x_i	the i^{th} component of the vector $\mathbf{x}$
$\mathbf{x}^{\mathrm{T}}$	transpose of vector
$\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n]$	a matrix
$\mathbf{a}_i$	the i^{th} vector of the matrix $\mathbf{A}$
a_{ij}	the i, j^{th} component of the matrix $\mathbf{A}$
$\mathbf{A}^{\mathrm{T}}$	transpose of matrix
$\mathbf{A}^{-1}$	matrix inverse
$\mathbf{A}^{+}$	matrix pseudo-inverse
$\mathbf{A}^{\frac{1}{2}}$	matrix root
$\nabla_{\theta_i} f$	derivative with respect to parameter θ_i
$\nabla_{\boldsymbol{\theta}} f$	derivative with respect to parameters θ_i
$\frac{\partial f}{\partial q}$	partial derivative
$p(\mathbf{x})$	probability density of $\mathbf{x}$
$E\{\mathbf{x}\}$	expectation of $\mathbf{x}$
$\bar{\mathbf{x}} = \langle \mathbf{x} \rangle$	sample average of $\mathbf{x}$

As symbols in this diploma thesis, the following symbols are used in several sections:

Symbol	Description
t	time
$\mathbf{x}, \dot{\mathbf{x}}, \ddot{\mathbf{x}}$	task space position, velocity, acceleration
$\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}$	joint space position, velocity, acceleration
$\mathbf{s}_t$	state (at time t)
$\mathbf{a}_t$	action (at time t)
$\mathbf{s}_{1:T+1}$	series of states $\mathbf{s}_t$ with $t \in \{1, 2, \ldots, T+1\}$
$\mathbf{a}_{1:T}$	series of actions $\mathbf{a}_t$ with $t \in \{1, 2, \ldots, T\}$
$\boldsymbol{\tau} = [\mathbf{s}_{1:T+1}, \mathbf{a}_{1:T}]$	rollout, episode, trial
T	rollout length
$\pi(\mathbf{a}_t \vert \mathbf{s}_t, t)$	policy
$\mathbb{T}$	set of all possible paths
r_t	reward (at time t)
$\mathbf{r}_{1:T}$	series of rewards r_t with $t \in \{1, 2, \ldots, T\}$
$R(\tau)$	return
$J(\boldsymbol{\theta})$	expected return
$D(p \Vert q)$	Kullback-Leibler divergence
$Q^{\pi}(\mathbf{s}, \mathbf{a}, t)$	value function of policy π
$\mathbf{F}(\boldsymbol{\theta})$	Fisher information matrix
ε	exploration
H	number of rollouts
n	index of parameter
k	index of iteration
$\mathbf{g} = \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$	gradient
α	update rate

Symbol	Description
$\boldsymbol{\psi}(\cdot)$	weights
$\mathbf{c}$	centers
$\mathbf{h}$	widths
l	length
m	mass
g	gravity
$\mathbf{F}$	forces
Δt	time step