Policy Learning for Motor Skills

Jan Peters[1,2] and Stefan Schaal[2,3]

[1] Max-Planck Institute for Biological Cybernetics, Spemannstr. 32, 72074 Tübingen
[2] University of Southern California, Los Angeles, CA 90089, USA
[3] ATR Computational Neuroscience Laboratory, Soraku-gun Kyoto 619-0288, Japan

Abstract. Policy learning which allows autonomous robots to adapt to novel situations has been a long standing vision of robotics, artificial intelligence, and cognitive sciences. However, to date, learning techniques have yet to fulfill this promise as only few methods manage to scale into the high-dimensional domains of manipulator robotics, or even the new upcoming trend of humanoid robotics, and usually scaling was only achieved in precisely pre-structured domains. In this paper, we investigate the ingredients for a general approach policy learning with the goal of an application to motor skill refinement in order to get one step closer towards human-like performance. For doing so, we study two major components for such an approach, i.e., firstly, we study policy learning algorithms which can be applied in the general setting of motor skill learning, and, secondly, we study a theoretically well-founded general approach to representing the required control structures for task representation and execution.

1 Introduction

Despite an increasing number of motor skills exhibited by manipulator and humanoid robots, the general approach to the generation of such motor behaviors has changed little over the last decades [15]. The roboticist models the task as accurately as possible and uses human understanding of the required motor skills in order to create the desired robot behavior as well as to eliminate all uncertainties of the environment. In most cases, such a process boils down to recording a desired trajectory in a pre-structured environment with precisely placed objects. If inaccuracies remain, the engineer creates exceptions using human understanding of the task. While such highly engineered approaches are feasible in well-structured industrial or research environments, it is obvious that if robots should ever leave factory floors and research environments, we will need to reduce or eliminate the strong reliance on hand-crafted models of the environment and the robots exhibited to date. Instead, we need a general approach which allows us to use compliant robots designed for interaction with less structured and uncertain environments in order to reach domains outside industry. Such an approach cannot solely rely on human knowledge but instead has to be acquired and adapted from data generated both by human demonstrations of the skill as well as trial and error of the robot.

The tremendous progress in machine learning over the last decades offers us the promise of less human-driven approaches to motor skill acquisition. However, despite offering the most general way of thinking about data-driven acquisition of motor skills, generic machine learning techniques, which do not rely on an understanding of motor

M. Ishikawa et al. (Eds.): ICONIP 2007, Part II, LNCS 4985, pp. 233–242, 2008.

systems, often do not scale into the domain of manipulator or humanoid robotics due to the high domain dimensionality. Therefore, instead of attempting an unstructured, monolithic machine learning approach to motor skill aquisition, we need to develop approaches suitable for this particular domain with the inherent problems of task representation, learning and execution addressed separately in a coherent framework employing a combination of imitation, reinforcement and model learning in order to cope with the complexities involved in motor skill learning. The advantage of such a concerted approach is that it allows the separation of the main problems of motor skill acquisition, refinement and control. Instead of either having an unstructured, monolithic machine learning approach or creating hand-crafted approaches with pre-specified trajectories, we are capable of aquiring skills, represented as policies, from demonstrations and refine them using trial and error. Using learning-based approaches for control, we can achieve accurate control without needing accurate models of the complete system.

2 Learning of Motor Skills

The principal objective of this paper is to find the foundations for a general framework for representing, learning and executing motor skills for robotics. As can be observed from this question, the major goal of this paper requires three building blocks, i.e., (i) appropriate representations for movements, (ii) learning algorithms which can be applied to these representations and (iii) a transformation which allows the execution of the kinematic policies in the respective task space on robots.

2.1 Essential Components

We address the three essential components, i.e., representation, learning and execution. In this section, we briefly outline the underlying fundamental concepts.

Representation. For the representation of motor skills, we can rely on the insight that humans, while being capable of performing a large variety of complicated movements, restrict themselves to a smaller amount of primitive motions [14]. As suggested by Ijspeert et al. [4], such primitive movements (or basic skills) can be represented by nonlinear dynamic systems. We can represent these in the differential constraint form given by $\boldsymbol{A}_{\boldsymbol{\theta}_i}(\boldsymbol{x}_i, \dot{\boldsymbol{x}}_i, t)\ddot{\boldsymbol{x}} = \boldsymbol{b}_{\boldsymbol{\theta}_i}(\boldsymbol{x}_i, \dot{\boldsymbol{x}}_i, t)$, where $i \in \mathbb{N}$ is the index of the motor primitive in a library of movements, $\boldsymbol{\theta}_i \in \mathbb{R}^L$ denote the parameters of the primitive i, t denotes time and $\boldsymbol{x}_i$,$\dot{\boldsymbol{x}}_i$,$\ddot{\boldsymbol{x}}_i \in \mathbb{R}^n$ denote positions, velocities and accelerations of the dynamic system, respectively. In the simplest case, $\boldsymbol{A}_{\boldsymbol{\theta}_i}$ could be an identity matrix and $\boldsymbol{b}_{\boldsymbol{\theta}_i}$ would be a desired task-spac acceleration. In more complicated cases, it could implicitly describe the task, see [8]. Note, that this dynamic system describes a task in its task space and *not necessarily* in the joint-space of the robot (which we denote by $\boldsymbol{q}$).

Learning. Learning basic motor skills[1] is achieved by adapting the parameters $\boldsymbol{\theta}_i$ of motor primitive i. The high dimensionality of our domain prohibits the exploration of the complete space of all admissible motor behaviors, rendering the application of

[1] Learning by sequencing and parallelization of the motor primitives (also referred to as basic skills) will be treated in future work.

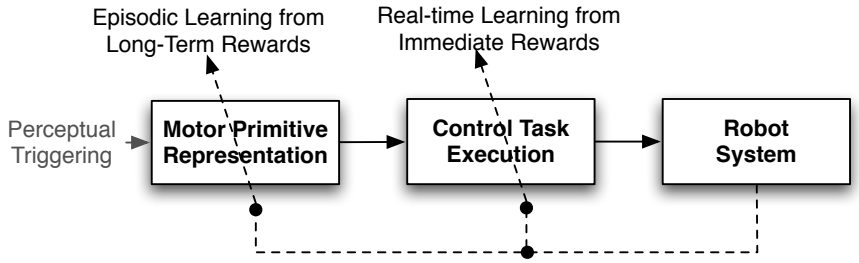

Fig. 1. This figure illustrates our general approach to motor skill learning by dividing it into motor primitive and a motor control component. For the task execution, fast policy learning methods based on observable error need to be employed while the task learning is based on slower episodic learning. The motor primitive yields a kinematic reference signal while the control task yields a motor command.

machine learning techniques which require exhaustive exploration impossible. Instead, we have to rely on a combination of supervised and reinforcement learning in order to aquire motor skills where the supervised learning is used in order to obtain the initialization of the motor skill while reinforcement learning is used in order to improve it. Therefore, the aquisition of a novel motor task consists out of two phases,i.e., the 'learning robot' attempts to reproduce the skill acquired through supervised learning and improve the skill from experience by trial-and-error, i.e., through reinforcement learning.

Execution. The execution of motor skills adds another level of complexity. It requires that a mechanical system $\boldsymbol{u} = \boldsymbol{M}(\boldsymbol{q}, \dot{\boldsymbol{q}}, t)\ddot{\boldsymbol{q}} + \boldsymbol{F}(\boldsymbol{q}, \dot{\boldsymbol{q}}, t)$, with a kinematic mapping to the task $\boldsymbol{x}_i = \boldsymbol{f}_i(\boldsymbol{q}, \dot{\boldsymbol{q}}, t)$ can be forced to execute each motor primitive $\boldsymbol{A}_i\ddot{\boldsymbol{x}}_i = \boldsymbol{b}_i$ in order to fulfill the skill. Here, $\boldsymbol{M}$ denotes the inertia matrix and $\boldsymbol{F}$ Coriolis, centrifugal and gravitational forces. The motor primitive can be viewed as a mechanical constraint acting upon the system, enforced through accurate computation of the required forces based on analytical models. However, in most cases it is very difficult to obtain accurate models of the mechanical system. Therefore it can be more suitable to find a policy learning approach which replaces the control law based on the hand-crafted rigid body model. In this paper, we will follow this approach which forms the basis for understanding motor skill learning.

2.2 Resulting Approach

As we have outlined during the discussion of our objective and its essential components, we require an appropriate general motor skill framework which allows us to separate the desired task-space movement generation (represented by the motor primitives) from movement control in the respective actuator space. Based on the understanding of this transformation from an analytical point of view on robotics, we present a learning framework for task execution in operational space. For doing so, we have to consider two components, i.e., we need to determine how to learn the desired behavior

represented by the motor primitives as well as the execution represented by the transformation of the motor primitives into motor commands. We need to develop scalable learning algorithms which are both appropriate and efficient when used with the chosen general motor skill learning architecture. Furthermore, we require algorithms for fast immediate policy learning for movement control based on instantly observable rewards in order to enable the system to cope with real-time improvement during the execution. The learning of the task itself on the other hand requires the learning of policies which define the long-term evolution of the task, i.e., motor primitives, which are learned on a trial-by-trial basis with episodic improvement using a teacher for demonstration and reinforcement learning for self-improvement. The resulting general concept underlying this paper is illustrated in Figure 1.

The resulting approach is related to approaches in neuroscientific models. It allows relating to both the the optimization based approaches (which have resulted in models like minimum jerk or minimum-torque change) as well as as to dynamic systems approaches (e.g., the VITE-FLETE model), see [13] for further information.

3 Policy Learning Approaches for Motor Skills

As outlined before, we need two different styles of policy learning algorithms, i.e., methods for long-term reward optimization and methods for immediate improvement. We can unify this goal by stating a cost function

$$J(\boldsymbol{\theta}) = \int_{\mathbb{T}} p_{\boldsymbol{\theta}}(\boldsymbol{\tau})\, r(\boldsymbol{\tau})\, d\boldsymbol{\tau}, \tag{1}$$

where $\boldsymbol{\tau}$ denotes a path, e.g., $\boldsymbol{\tau} = [\boldsymbol{x}_{1:n}, \boldsymbol{u}_{1:n}]$ with states $\boldsymbol{x}_{1:n}$ and actions $\boldsymbol{u}_{1:n}$, $r(\boldsymbol{\tau})$ denotes the reward along the path, e.g., $r(\boldsymbol{\tau}) = \sum_{t=1}^{n} \gamma^t r_t$ and $p_{\boldsymbol{\theta}}(d\boldsymbol{\tau})$ denotes the path probability density $p_{\boldsymbol{\theta}}(d\boldsymbol{\tau}) = p(\boldsymbol{x}_1) \prod_{t=1}^{n-1} p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)\, \pi(\boldsymbol{u}_t|\boldsymbol{x}_t; \boldsymbol{\theta})$ with a first-state distribution $p(\boldsymbol{x}_1)$, a state transition $p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)$ and a policy $\pi(\boldsymbol{u}_t|\boldsymbol{x}_t; \boldsymbol{\theta})$. Note, that $p_{\boldsymbol{\theta}}(\boldsymbol{\tau})\, r(\boldsymbol{\tau})$ is an improper distribution, i.e., does not integrate to 1. The policy $\pi(\boldsymbol{u}_t|\boldsymbol{x}_t; \boldsymbol{\theta})$ is the function which we intend to learn by optimizing its parameters $\boldsymbol{\theta} \in \mathbb{R}^N$. Many policy learning algorithms have started optimize this cost function, including policy gradient methods [1], actor-critic methods [16,6], the Natural Actor-Critic [10,11,12] and Reward-Weighted Regression [9]. In the remainder of this section, we will sketch a unified approach to policy optimization which allows the derivation of all of the methods above from the variation of a single cost function. This section might appear rather abstract in comparison to the rest of the paper; however, it contains major novelties as it allows a coherent treatment of many previous and future approaches.

3.1 Bounds for Policy Updates

In this section, we will look at two problems in policy learning, i.e., an upper bound and a lower bound on policy improvements. The upper bound outlines why a greedy operator is not a useful solution while the lower bound will be used to derive useful policy updates.

Upper Bound on Policy Improvements. In the stochastic programming community, it is well-known that the greedy approach to policy optimization suffers from the major drawback that it can return only a biassed solution. This drawback can be formalized straighforwardly by showing that if we optimize $J(\boldsymbol{\theta})$ and approximate it by samples, e.g., by $\hat{J}_S(\boldsymbol{\theta}) = \sum_{s=1}^{S} p_{\boldsymbol{\theta}}(\boldsymbol{\tau}_s) r(\boldsymbol{\tau}_s) \approx J(\boldsymbol{\theta})$, we obtain the fundamental relationship

$$E\{\max_{\boldsymbol{\theta}} \hat{J}_S(\boldsymbol{\theta})\} \geq \max_{\boldsymbol{\theta}} E\{\hat{J}_S(\boldsymbol{\theta})\}, \tag{2}$$

which can be shown straightforwardly by first realizing the that the maximum is always larger than any member of a sample. Thus, a subsequent expectation will not change this fact nor the subsequent optimization of the lower bound. Thus, a policy which is optimized by doing a greedy step in parameter space is guaranteed to be biased in the presence of errors with a bias of $b_S(\boldsymbol{\theta}) = E\{\max_{\boldsymbol{\theta}} \hat{J}_S(\boldsymbol{\theta})\} - \max_{\boldsymbol{\theta}} E\{\hat{J}_S(\boldsymbol{\theta})\} \geq 0$. However, we can also show that the bias decreases over the number of samples, i.e., $b_S(\boldsymbol{\theta}) \geq b_{S+1}(\boldsymbol{\theta})$, and converges to zero for infinite samples, i.e., $\lim_{S\to\infty} b_S(\boldsymbol{\theta}) = 0$ [7]. This optimization bias illustrates the deficiencies of the greedy operator: for finite data any policy update is problematic and can result into unstable learning processes with oscillations, divergence, etc as frequently observed in the reinforcement learning community [2,1].

Lower Bound on Policy Improvements. In other branches of machine learning, the focus has been on lower bounds, e.g., in Expectation-Maximization (EM) algorithms. 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. Surprisingly, the lower bounds in supervised learning can be transferred with ease. For doing so, we look at the scenario (suggested in [3]) that we have a policy $\boldsymbol{\theta}'$ and intend to match the path distribution generated by this policy to the success weighted path distribution, then we intend to minimize the distance between both distributions, i.e., $D(p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}) || p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) r(\boldsymbol{\tau}))$. Surprisingly, this results into a lower bound using Jensen's inequality and the convexity of the logarithm function. This results into

$$\log J(\boldsymbol{\theta}') = \log \int \frac{p_{\boldsymbol{\theta}}(\boldsymbol{\tau})}{p_{\boldsymbol{\theta}}(\boldsymbol{\tau})} p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}) r(\boldsymbol{\tau}) d\boldsymbol{\tau}, \tag{3}$$

$$\geq \int p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) r(\boldsymbol{\tau}) \log \frac{p_{\boldsymbol{\theta}'}(\boldsymbol{\tau})}{p_{\boldsymbol{\theta}}(\boldsymbol{\tau})} d\boldsymbol{\tau} \propto -D(p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}) || p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) r(\boldsymbol{\tau})), \tag{4}$$

where $D(p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}) || p_{\boldsymbol{\theta}}(\boldsymbol{\tau})) = \int p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) \log(p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) / p_{\boldsymbol{\theta}'}(\boldsymbol{\tau})) d\boldsymbol{\tau}$ is the Kullback-Leibler divergence, i.e., a distance measure for probability distributions. With other words, we have the lower bound $J(\boldsymbol{\theta}') \geq \exp(D(p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}) || p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) r(\boldsymbol{\tau})))$, and we can minimize

$$J_{\mathrm{KL}} = D(p_{\boldsymbol{\theta}'}(\boldsymbol{\tau}) || p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) r(\boldsymbol{\tau})) = \int p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) r(\boldsymbol{\tau}) \log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) r(\boldsymbol{\tau})}{p_{\boldsymbol{\theta}'}(\boldsymbol{\tau})} d\boldsymbol{\tau} \tag{5}$$

without the problems which have troubled the reinforcement learning community when optimizing the upper bound as we are guaranteed to improve the policy. However, in many cases, we might intend to punish divergence from the previous solution. In this case, we intend to additionally control the distance which we move away from our

previous policy, e.g., minimize the term $J_+ = -D\left(p_{\boldsymbol{\theta}}\left(\boldsymbol{\tau}\right)||p_{\boldsymbol{\theta}'}\left(\boldsymbol{\tau}\right)\right)$. We can combine these into a joint cost function

$$J_{\mathrm{KL}+} = J_{\mathrm{KL}} + \lambda J_+, \tag{6}$$

where $\lambda \in \mathbb{R}^+$ is a positive punishment factor with $0 \leq \lambda \leq J(\boldsymbol{\theta})$. Note that the exchange of the arguments is due to the fact that the Kullback-Leibler divergence is unsymmetric. This second term will play an important rule as both baselines and natural policy gradients are a directly result of it. The proper determination of λ is non-trivial and depends on the method. E.g., in policy gradients, this becomes the baseline.

3.2 Resulting Approaches for Policy Learning

We now proceed into deriving three different methods for lower bound optimization, i.e., policy gradients, the natural actor-critic and reward-weighted regression. All three of these can be derived from this one perspective.

Policy Gradients Approaches. It has recently been recognized that policy gradient methods [2,1] do not suffer from the drawbacks of the greedy operator and, thus, had a large revival in recent years. We can derive policy gradient approaches straightforwardly from this formulation using the steepest descent of the first order taylor extension

$$\boldsymbol{\theta}' = \boldsymbol{\theta} + \alpha(\boldsymbol{\nabla} J_{\mathrm{KL}} - \lambda \boldsymbol{\nabla} J_+) \tag{7}$$

$$= \boldsymbol{\theta} + \alpha \int p_{\boldsymbol{\theta}}\left(\boldsymbol{\tau}\right)\left(r\left(\boldsymbol{\tau}\right) - \lambda\right) \boldsymbol{\nabla} \log p_{\boldsymbol{\theta}'}\left(\boldsymbol{\tau}\right) d\boldsymbol{\tau}, \tag{8}$$

where α is a learning rate. This is only true as for the first derivative $\boldsymbol{\nabla} D\left(p_{\boldsymbol{\theta}}\left(\boldsymbol{\tau}\right)||p_{\boldsymbol{\theta}'}\left(\boldsymbol{\tau}\right)\right) = \boldsymbol{\nabla} D\left(p_{\boldsymbol{\theta}'}\left(\boldsymbol{\tau}\right)||p_{\boldsymbol{\theta}}\left(\boldsymbol{\tau}\right)\right)$. The punishment factor from before simply becomes the baseline of the policy gradient estimator. As $\boldsymbol{\nabla} \log p_{\boldsymbol{\theta}'}\left(\boldsymbol{\tau}\right) = \sum_{t=1}^{n-1} \boldsymbol{\nabla} \log \pi(\boldsymbol{u}_t|\boldsymbol{x}_t;\boldsymbol{\theta})$, we obtain the straightforward gradient estimator also known as REINFORCE, policy gradient theorem or GPOMDP, for an overview see [1]. The punishment term only constrains the variance of the policy gradient estimate and vanishes as $\boldsymbol{\nabla} J_{\mathrm{KL}+} = \boldsymbol{\nabla} J_{\mathrm{KL}}$ for infinite data. However, this policy update can be shown to be rather slow [5,10,11,12].

Natural Policy Gradient Approaches. Suprisingly, the speed update can be improved significantly if we punish higher order terms of J_+, e.g., the second term of the taylor expansion yields

$$\boldsymbol{\theta}' = \mathrm{argmax}_{\boldsymbol{\theta}'}(\boldsymbol{\theta}' - \boldsymbol{\theta})^T(\boldsymbol{\nabla} J_{\mathrm{KL}} - \lambda \boldsymbol{\nabla} J_+) - \frac{1}{2}\lambda(\boldsymbol{\theta}' - \boldsymbol{\theta})^T \boldsymbol{\nabla}^2 J_+(\boldsymbol{\theta}' - \boldsymbol{\theta}) \tag{9}$$

$$= \lambda \left(\boldsymbol{\nabla}^2 J_+\right)^{-1} (\boldsymbol{\nabla} J_{\mathrm{KL}} - \lambda \boldsymbol{\nabla} J_+) = \lambda \boldsymbol{F}^{-1} g_1, \tag{10}$$

where $\boldsymbol{F} = \boldsymbol{\nabla}^2 D\left(p_{\boldsymbol{\theta}}\left(\boldsymbol{\tau}\right)||p_{\boldsymbol{\theta}'}\left(\boldsymbol{\tau}\right)\right) = \boldsymbol{\nabla}^2 D\left(p_{\boldsymbol{\theta}'}\left(\boldsymbol{\tau}\right)||p_{\boldsymbol{\theta}}\left(\boldsymbol{\tau}\right)\right) = \boldsymbol{\nabla}^2 J_+$ is also known as the Fisher information matrix and the resulting policy update $\boldsymbol{g}_2$ is known as the Natural Policy Gradient. Surprisingly, the second order term has not yet been expanded and no Natural second-order gradient approaches are known. Thus, this could potentially be a great topic for future research.

EM-Policy Learning. In a very special case, we can solve for the optimal policy parameters, e.g, for policy which are linear in the log-derivatives such as

$$\boldsymbol{\nabla} \log \pi(\boldsymbol{u}_t|\boldsymbol{x}_t;\boldsymbol{\theta}) = \boldsymbol{A}\left(\boldsymbol{x}_t,\boldsymbol{u}_t\right)\boldsymbol{\theta} + \boldsymbol{b}\left(\boldsymbol{x}_t,\boldsymbol{u}_t\right), \tag{11}$$

it is straightforward to derive an EM algorithm such as

$$\boldsymbol{\theta}' = \alpha^{-1}\beta, \tag{12}$$

$$\boldsymbol{\alpha} = \int p_{\boldsymbol{\theta}}\left(\boldsymbol{\tau}\right)\left(r\left(\boldsymbol{\tau}\right) - \lambda\right)\sum_{t=1}^{n}\boldsymbol{A}\left(\boldsymbol{x}_t,\boldsymbol{u}_t\right)d\boldsymbol{\tau}, \tag{13}$$

$$\boldsymbol{\beta} = \int p_{\boldsymbol{\theta}}\left(\boldsymbol{\tau}\right)\left(r\left(\boldsymbol{\tau}\right) - \lambda\right)\sum_{t=1}^{n} b\left(\boldsymbol{x}_t,\boldsymbol{u}_t\right)d\boldsymbol{\tau}. \tag{14}$$

This type of algorithms can result into very fast policy updates if applicable. It does not require a learning rate and is guaranteed to converge to at least a locally optimal solution.

3.3 Sketch of the Resulting Algorithms

Thus, we have developed two different classes of algorithms, i.e., the Natural Actor-Critic and the Reward-Weighted Regression.

Natural Actor-Critic. The Natural Actor-Critic algorithms [10,11] instantiations of the natural policy gradient previously described with a large or infinite horizon n. They are considered the fastest policy gradient methods to date and "the current method of choice" [1]. They rely on the insight that we need to maximize the reward while keeping the loss of experience constant, i.e., we need to measure the distance between our current path distribution and the new path distribution created by the policy. This distance can be measured by the Kullback-Leibler divergence and approximated using the Fisher information metric resulting in a natural policy gradient approach. This natural policy gradient has a connection to the recently introduced compatible function approximation, which allows to obtain the Natural Actor-Critic. Interestingly, earlier Actor-Critic approaches can be derived from this new approach. In application to motor primitive learning, we can demonstrate that the Natural Actor-Critic outperforms both finite-difference gradients as well as 'vanilla' policy gradient methods with optimal baselines.

Reward-Weighted Regression. In contrast to Natural Actor-Critic algorithms, the Reward-Weighted Regression algorithm [9] focuses on immediate reward improvement, i.e., $n = 1$, and employs an adaptation of the expectation maximization (EM) policy learning algorithm for reinforcement learning as previously described instead of a gradient based approach. The key difference here is that when using immediate rewards, we can learn from our actions directly, i.e., use them as training examples similar to a supervised learning problem with a higher priority for samples with a higher reward. Thus, this problem is a reward-weighted regression problem, i.e., it has a well-defined solution which can be obtained using established regression techniques. While

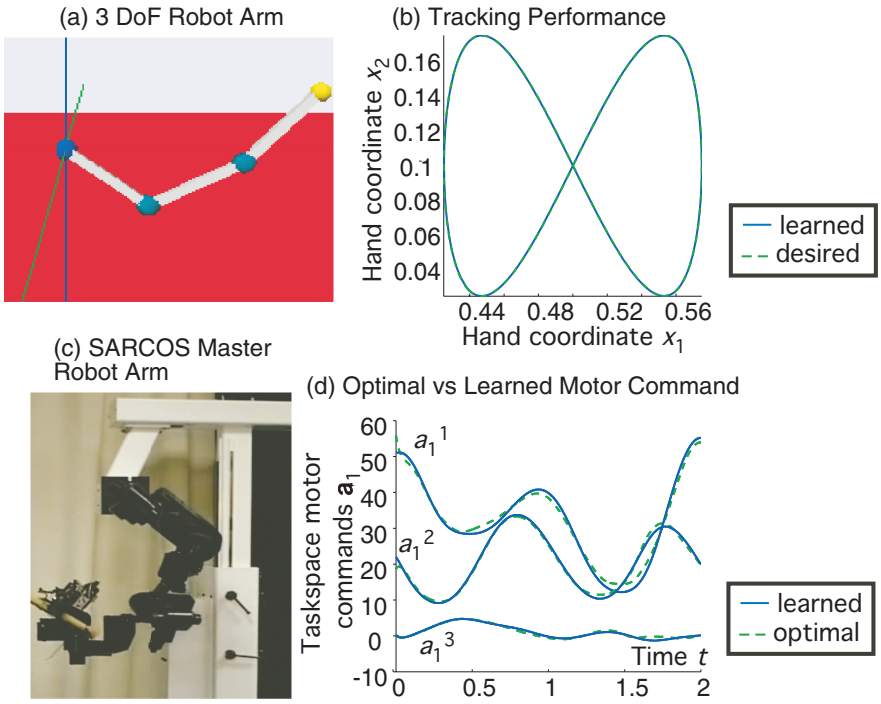

Fig. 2. Systems and results of evaluations for learning operational space control: (a) screen shot of the 3 DOF arm simulator, (c) Sarcos robot arm, used as simulated system and for actual robot evaluations in progress. (b) Tracking performance for a planar figure-8 pattern for the 3 DOF arm, and (d) comparison between the analytically obtained optimal control commands in comparison to the learned ones for one figure-8 cycle of the 3DOF arm.

we have given a more intuitive explanation of this algorithm, it corresponds to a properly derived maximization-maximization (MM) algorithm which maximizes a lower bound on the immediate reward similar to an EM algorithm. Our applications show that it scales to high dimensional domains and learns a good policy without any imitation of a human teacher.

4 Robot Application

The general setup presented in this paper can be applied in robotics using analytical models as well as the presented learning algorithms. The applications presented in this paper include motor primitive learning and operational space control.

4.1 Learning Operational Space Control

Operational space control is one of the most general frameworks for obtaining task-level control laws in robotics. In this paper, we present a learning framework for operational

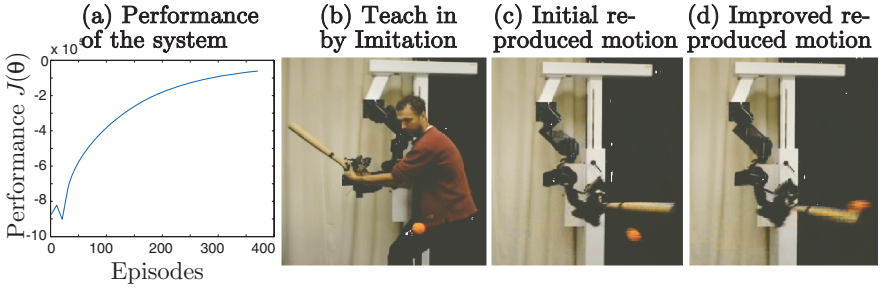

Fig. 3. This figure shows (a) the performance of a baseball swing task when using the motor primitives for learning. In (b), the learning system is initialized by imitation learning, in (c) it is initially failing at reproducing the motor behavior, and (d) after several hundred episodes exhibiting a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

space control which is a result of a reformulation of operational space control as a general point-wise optimal control framework and our insights into immediate reward reinforcement learning. While the general learning of operational space controllers with redundant degrees of freedom is non-convex and thus global supervised learning techniques cannot be applied straightforwardly, we can gain two insights, i.e., that the problem is locally convex and that our point-wise cost function allows us to ensure global consistency among the local solutions. We show that this can yield the analytically determined optimal solution for simulated three degrees of freedom arms where we can sample the state-space sufficiently. Similarly, we can show the framework works well for simulations of the both three and seven degrees of freedom robot arms as presented in Figure 2.

4.2 Motor Primitive Improvement by Reinforcement Learning

The main application of our long-term improvement framework is the optimization of motor primitives. Here, we follow essentially the previously outlined idea of acquiring an initial solution by supervised learning and then using reinforcement learning for motor primitive improvement. For this, we demonstrate both comparisons of motor primitive learning with different policy gradient methods, i.e., finite difference methods, 'vanilla' policy gradient methods and the Natural Actor-Critic, as well as an application of the most successful method, the Natural Actor-Critic to T-Ball learning on a physical, anthropomorphic SARCOS Master Arm, see Figure 3.

5 Conclusion

In conclusion, in this paper, we have presented a general framework for learning motor skills which is based on a thorough, analytically understanding of robot task representation and execution. We have introduced a general framework for policy learning which allows the derivation of a variety of novel reinforcement learning methods including the Natural Actor-Critic and the Reward-Weighted Regression algorithm. We demonstrate

the efficiency of these reinforcement learning methods in the application of learning to hit a baseball with an anthropomorphic robot arm on a physical SARCOS master arm using the Natural Actor-Critic, and in simulation for the learning of operational space with reward-weighted regression.

References

1. Aberdeen, D.: POMDPs and policy gradients. In: Proceedings of the Machine Learning Summer School (MLSS), Canberra, Australia (2006)
2. Aberdeen, D.A.: Policy-Gradient Algorithms for Partially Observable Markov Decision Processes. PhD thesis, Australian National Unversity (2003)
3. Dayan, P., Hinton, G.E.: Using expectation-maximization for reinforcement learning. Neural Computation 9(2), 271–278 (1997)
4. Ijspeert, A., Nakanishi, J., Schaal, S.: Learning attractor landscapes for learning motor primitives. In: Becker, S., Thrun, S., Obermayer, K. (eds.) Advances in Neural Information Processing Systems, vol. 15, pp. 1547–1554. MIT Press, Cambridge (2003)
5. Kakade, S.A.: Natural policy gradient. In: Advances in Neural Information Processing Systems, Vancouver, CA, vol. 14 (2002)
6. Konda, V., Tsitsiklis, J.: Actor-critic algorithms. Advances in Neural Information Processing Systems 12 (2000)
7. Peters, J.: The bias of the greedy update. Technical report, University of Southern California (2007)
8. Peters, J., Mistry, M., Udwadia, F., Cory, R., Nakanishi, J., Schaal, S.: A unifying methodology for the control of robotic systems. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Edmonton, Canada (2005)
9. Peters, J., Schaal, S.: Learning operational space control. In: Proceedings of Robotics: Science and Systems (RSS), Philadelphia, PA (2006)
10. Peters, J., Vijayakumar, S., Schaal, S.: Reinforcement learning for humanoid robotics. In: Proceedings of the IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS), Karlsruhe, Germany (September 2003)
11. Peters, J., Vijayakumar, S., Schaal, S.: Natural actor-critic. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) ECML 2005. LNCS (LNAI), vol. 3720, pp. 280–291. Springer, Heidelberg (2005)
12. Richter, S., Aberdeen, D., Yu, J.: Natural actor-critic for road traffic optimisation. In: Schoelkopf, B., Platt, J.C., Hofmann, T. (eds.) Advances in Neural Information Processing Systems, vol. 19, MIT Press, Cambridge (2007)
13. Schaal, S.: Dynamic movement primitives - a framework for motor control in humans and humanoid robots. In: Proceedings of the International Symposium on Adaptive Motion of Animals and Machines (2003)
14. Schaal, S., Ijspeert, A., Billard, A.: Computational approaches to motor learning by imitation. In: Frith, C.D., Wolpert, D. (eds.) The Neuroscience of Social Interaction, pp. 199–218. Oxford University Press, Oxford (2004)
15. Sciavicco, L., Siciliano, B.: Modeling and control of robot manipulators. MacGraw-Hill, Heidelberg (2007)
16. Sutton, R.S., McAllester, D., Singh, S., Mansour, Y.: Policy gradient methods for reinforcement learning with function approximation. In: Solla, S.A., Leen, T.K., Mueller, K.-R. (eds.) Advances in Neural Information Processing Systems (NIPS), Denver, CO, MIT Press, Cambridge (2000)