Robot Weightlifting By Direct Policy Search

Michael T. Rosenstein and Andrew G. Barto
Department of Computer Science
University of Massachusetts
Amherst, MA 01003-4610
{mtr, barto}@cs.umass.edu

Abstract

This paper describes a method for structuring a robot motor learning task. By designing a suitably parameterized policy, we show that a simple search algorithm, along with biologically motivated constraints, offers an effective means for motor skill acquisition. The framework makes use of the robot counterparts to several elements found in human motor learning: imitation, equilibrium-point control, motor programs, and synergies. We demonstrate that through learning, coordinated behavior emerges from initial, crude knowledge about a difficult robot weightlifting task.

1 Introduction

Humans exhibit remarkable proficiency at complex motor tasks such as handwriting, juggling, and machine assembly. One way to characterize our success with such tasks is that we have a mastery of redundant degrees of freedom, cf. Bernstein [1967]. The ease with which humans achieve motor coordination contrasts sharply with that of robots, where "simple" motor tasks such as walking and throwing pose challenges to roboticists and artificial intelligence researchers. The typical solution of path planning and trajectory tracking works well for highly structured problems but not for situations with complex, nonlinear dynamics, with inadequate models, and with no detailed knowledge of the best solution. In this paper we examine a form of trial-and-error learning as a means for motor skill acquisition. Our results involve a simulated, three-link robotic arm that learns to raise heavy weights through efficient, timely use of its actuators.

Algorithms for trial-and-error learning typically fall near one of two extremes: those that take advantage of structure in the problem, and those that take advantage of structure in the solution. We have in mind the reinforcement learning problem where an agent interacts repeatedly with its environment and attempts to learn an optimal policy, i.e., a mapping from states to actions that maximizes some performance criterion. Algorithms that focus on the solution often make direct adjustments of the current policy, whereas methods that focus on the problem tend to build an intermediate representation, such as a value function, that forms the basis for policy improvement. In any case, the distinguishing features of trial-and-error learning are the exploratory activity and the simple evaluative feedback that reinforces successful behavior.

Reinforcement learning algorithms such as TD(λ) [Sutton, 1988] and Q-learning [Watkins and Dayan, 1992] are particularly well-suited to take advantage of structure in the problem. Such algorithms use a value function to capture regularities in the observed rewards and state transitions that implicitly define the task. However, reinforcement learning algorithms typically ignore structure in the solution by treating the policy as a featureless mapping from states to actions. Extensions to the basic reinforcement learning framework, such as function approximation, e.g., [Tsitsiklis and Van Roy, 1997], variable-resolution methods, e.g., [Moore and Atkeson, 1995], factored MDPs, e.g., [Boutilier *et al.*, 2000], and semi-Markov decison problems, e.g., [Sutton *et al.*, 1999], all constrain the form of the policy and, therefore, add strucutre to the solution. Nevertheless, with each of these extensions the focus remains on the value function rather than on the policy.

At the other extreme, methods for direct policy search, e.g., [Anderson, 2000; Baird and Moore, 1999; Baxter and Bartlett, 2000; Moriarty *et al.*, 1999], do away with intermediate representations of the policy and take advantage of constraints placed on the solution. For instance, evolutionary algorithms [Moriarty *et al.*, 1999] reinforce regularities in the parameterized policy (the "genetic material") by restricting "reproduction" to the fittest members of the population. And direct methods for function optimization [Swann, 1972] make the most of continuity by biasing their search toward promising regions of parameter space. However, such methods ignore structure in sequential decision problems by making policy adjustments on a trial-by-trial basis after long-term rewards are known, rather than on a per-action basis.

Both direct policy search and value-based methods often start with little or no prior structure and rely on extensive exploration to gather the information needed to build an optimal policy. Both approaches can benefit from domain knowledge. In this paper we utilize an algorithm for direct policy search because of its simplicity, not because we believe such methods are generally superior to value-based algorithms. Our goal is to show that biologically motivated structure can make trial-and-error learning an effective approach for a particular

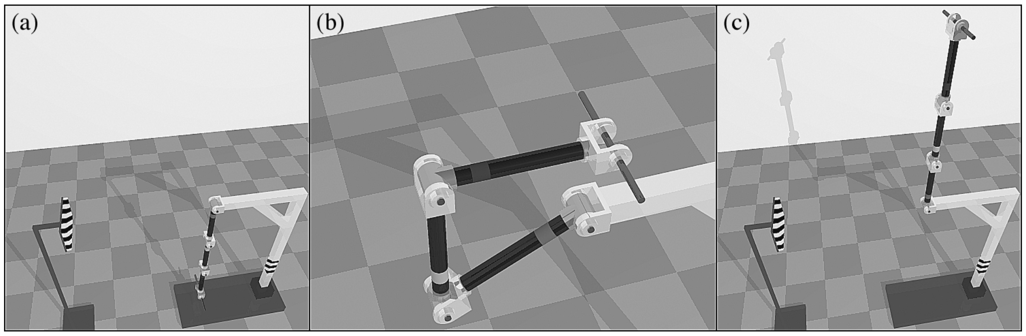

Figure 1: Simulated three-link robotic arm in several configurations with no payload: (a) start, (b) via point, and (c) goal. At the via point, the joint angles are -150, -120, and -90 degrees (proximal to distal).

class of control problems. Although we are certainly not the first to look to biology for inspiration with regard to machine learning and control, this work is novel in its combination of motor strategies for developing a coordinated robot.

2 Robot Weightlifting

By exploiting the dynamics of the task—gravity, inertia, elasticity, and so on—Olympic weightlifters solve a difficult coordination problem and raise more than twice their bodyweight over their heads. Our simulated robotic weightlifter, while loosely based on Olympic weightlifters, was inspired by the numerous acrobot-like examples studied by the machine learning and control engineering communities. Figure 1 shows the robot in several configurations. The task is to lift a payload from the straight-down, stable equilibrium to the straight-up, unstable equilibrium. Limited torque at each joint restricts the class of feasible solutions to those that execute a coordinated swing, i.e., those that exploit momentum and intersegmental dynamics. Kinematic redundancy, joint constraints, and obstacles (the striped object as well as the robot and its support structure) also complicate the task.

The robotic arm was simulated as a three-link frictionless pendulum with each link having length 1 m and and mass 1 kg. At each joint, torque was limited to $[-50,50]$ Nm, and at the middle and distal joints the range of motion was confined to $[-150,150]$ degrees. The equations of motion were generated iteratively using the Newton-Euler method [Walker and Orin, 1982] and solved numerically by Euler integration with a step size of 0.001 s. The robot was motionless at the start, and the goal was defined as a six-dimensional hypercube with velocity limits of ± 28.65 deg/s (± 0.5 rad/sec) and with position limits of ± 5 degrees centered on the goal configuration. At the end of each trial, i.e., when the robot's state entered the goal region, performance was quantified as the total integrated torque magnitude, and on those trials where the lifter exceeded its range of motion, contacted an obstacle, or failed to reach the goal within five seconds, the trial was terminated with the maximum cost of 500 Nm$\cdot$s. In summary, the task was designed as a minimum-effort optimal control problem with both torque and kinematic constraints.

Optimal control theory provides a number of techniques for solving dynamic optimization problems such as our weightlifting task. For instance, Wang *et al.* [1999] used a gradient-based method starting from an initial feasible path to more than triple the recommended payload capacity of a Puma industrial robot. However, such methods usually require precise models (or system identification) and protracted offline computation before the robot attempts its first lift. In this paper we make no assumptions about the availability of a detailed model. Instead, we borrow several human motor learning strategies as a way to add the needed structure to an otherwise difficult machine learning problem.

3 Structured Policy Parameterization

Most research on human motor coordination has focused on control instead of learning, and so relatively little is known about the *mechanisms* of human motor learning. This leaves us, the robot designers, in a position to select from many available artificial intelligence and adaptive control techniques. Below, we describe a simple form of trial-and-error learning. However, our focus is not the choice of a specific algorithm but rather the motor control strategies that place constraints on that algorithm. In particular, we *design* a biologically motivated, parameterized policy that is easily seeded with prior knowledge and easily adapted through learning.

3.1 Biological Motivation

Our proposed solution begins with a sequence of *via points*, i.e., desired configurations for the robotic arm. (A similar approach was used by Morimoto and Doya [1998] for a robot stand-up task.) We regard each via point as the specification for a movement primitive that converges to the corresponding manipulator configuration. Such primitives are suggestive of an equilibrium-point style of motor control, e.g., [Feldman, 1986], whereby the endpoint of a movement is programmed and the spring-like properties of the muscles ensure convergence to that endpoint. The primary benefits of equilibrium-point control are that complex limb dynamics can be ignored and that higher motor centers need not be involved in the detailed activation patterns of numerous actuators. The drawback is that to explain complicated movements, equilibrium-point approaches give up their simplicity by requiring a *virtual trajectory*, or sequence of equilibrium points that induce the desired movement but are not targets for convergence.

For the weightlifting task, one possible implementation of equilibrium-point control involves the use of a single movement primitive that brings the manipulator to the goal configuration. Indeed, with no payload and with no obstacle a simple linear feedback controller accomplishes the task, although this solution fails with payloads as small as 0.5 kg. Instead, we add a second movement primitive for the via point shown in Figure 1b. This via point represents the knowledge that certain configurations avoid the leftmost obstacle and also reduce the effective moment arm for the proximal joint. We assume the via point is obtained through imitation of a successful lift (cf. Schaal [1999]) or through instruction from a coach—a human programmer in our case. The via point is intended to convey crude path information, with no detailed knowledge of any successful trajectory.

Convergence first to the via point and then to the goal, extends the payload capacity to about 2 kg, beyond which adaptation is necessary. Thus, for the robotic weightlifter the advantage of equilibrium-point control is also its drawback: The robot's intrinsic dynamics are ignored for their difficulties as well as their benefits. Rather than turn to virtual trajectories as a way to exploit dynamics, we use a small number of movement primitives that act as the starting point for learning. The result is a hierarchical motor program that runs three feedback controllers: one for the via point and one for the goal, both adjustable, followed by another, fixed controller to regulate locally about the goal. This converts the closed-loop equilibrium-point solution to a "ballpark" open-loop solution [Greene, 1972; Schmidt, 1975] that handles minor errors at a lower, closed-loop level of control.

3.2 Implementation

To build the initial motor program, we first construct two proportional-derivative (PD) controllers, PD_1 and PD_2:

$$\text{PD}_i : \quad \boldsymbol{\tau}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) = \mathbf{W}_i[K_p(\boldsymbol{\theta}_i^* - \boldsymbol{\theta}) - K_v\dot{\boldsymbol{\theta}}], \tag{1}$$

where $\boldsymbol{\theta}$ and $\dot{\boldsymbol{\theta}}$ are the joint positions and velocities, respectively, and $\boldsymbol{\tau} \in \Re^3$ is a vector of joint torques subject to saturation at the ± 50 Nm torque limit. In Eq. (1) $\mathbf{W}_i$ is a 3x3 gain matrix, $\boldsymbol{\theta}_i^*$ is the target equilibrium point, and $K_p = 1000$ Nm·rad^{-1} and $K_v = 250$ Nm·s·rad^{-1} are the nominal proportional and derivative gains, respectively. The target equilibrium points, $\boldsymbol{\theta}_1^*$ and $\boldsymbol{\theta}_2^*$, are initialized to the via point and goal configuration, respectively. Both $\mathbf{W}_1$ and $\mathbf{W}_2$ are initialized to the identity matrix, and so each PD controller initially acts as three uncoupled, scalar-output servomechanisms (one for each joint).

Next, the robot executes an "imitation" trial by running to convergence PD_1 followed by PD_2. (A threshold test on the position error establishes convergence.) At convergence we record t_1^* and t_2^*—the time elapsed for the corresponding controller since the start of the trial. Together t_1^* and t_2^* mark the switching times for the open-loop level of control. In particular, the program runs PD_1 from the start of each new trial (time $t = 0$) until $t = t_1^*$, then a switch is made to PD_2 which runs until $t = t_2^*$, followed by the third controller, PD_3, that runs until the trial terminates. PD_3 is a fixed copy of PD_2 that increases the flexibility of the learning system while providing convergence guarantees for movements close to the goal. However, PD_3 plays no role during the imitation trial.

The open-loop level of the motor program has two free parameters, the switching times, that we adjust by trial-and-error learning (described shortly). Together, PD_1 and PD_2 have 24 free parameters, six from $\boldsymbol{\theta}_i^*$ and 18 from $\mathbf{W}_i$, that we adapt with the same learning algorithm. As a form of shaping, we increase the payload at a nominal rate of 0.25 kg every 250 trials of learning. And with each new payload, learning proceeds in two phases: a shorter phase (50 trials) that adjusts the open-loop timing of the motor program, followed by a longer phase (200 trials) that adjusts the lower-level PD controllers.

3.3 Direct Policy Search

Table 1 summarizes the *simple random search* (SRS) algorithm developed for the robot weightlifting task. The algorithm performs random search in a K-dimensional parameter space, centered at a base point, $\mathbf{x}$. Perturbations, $\Delta\mathbf{x}$, are normally distributed with zero mean and standard deviation equal to the search size, σ. Each test point $(\mathbf{x} + \Delta\mathbf{x})$ is evaluated and the best observed point is kept as the algorithm's return value. For the weightlifting task, the evaluation function gives the total effort during a simulated trial with the new parameter settings. Updates to the base point are made by taking a step toward the most recent test point with probability β or toward the best observed point with probability $1 - \beta$. Thus, β provides an easy adjustment for the exploration-exploitation tradeoff, with the extremes of a pure random walk ($\beta = 1$) and of movement along a rough estimate of the gradient ($\beta = 0$). Even with β set to zero, considerable exploration is possible for large values of σ, which decays by a factor γ after each iteration, to the minimum σ_{min}.

```
input
    initial point $\mathbf{x} \in \Re^K$
    step size $\alpha \in [0,1]$
    search strategy $\beta \in [0,1]$
    search size $\sigma \geq 0$
    search decay factor $\gamma \in [0,1]$
    minimum search size $\sigma_{min} \geq 0$
initialize
    $\mathbf{x}_{best} \leftarrow \mathbf{x}$
    $y_{best} \leftarrow \texttt{EVALUATE}(\mathbf{x}_{best})$
repeat
    1.  $\Delta\mathbf{x} \leftarrow \mathbf{N}(\mathbf{0}, \sigma)$
    2.  $y \leftarrow \texttt{EVALUATE}(\mathbf{x} + \Delta\mathbf{x})$
    3.  if $y < y_{best}$
    4.      $\mathbf{x}_{best} \leftarrow \mathbf{x} + \Delta\mathbf{x}$
    5.      $y_{best} \leftarrow y$
    6.  $\mathbf{x} \leftarrow \begin{cases} \mathbf{x} + \alpha \cdot \Delta\mathbf{x}, & \text{with prob. } \beta \\ \mathbf{x} + \alpha[\mathbf{x}_{best} - \mathbf{x}], & \text{with prob. } 1 - \beta \end{cases}$
    7.  $\sigma \leftarrow \texttt{max}(\gamma\sigma, \sigma_{min})$
until convergence or number of iterations too large
return $\mathbf{x}_{best}$
```

Table 1: The simple random search (SRS) algorithm. For the motor program, $\mathbf{x} \in \Re^2$ at the open-loop level of control and $\mathbf{x} \in \Re^{24}$ at the closed-loop level.

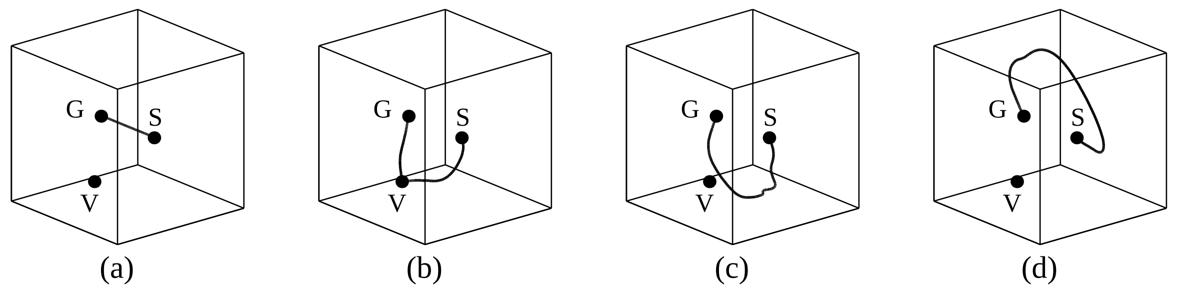

Figure 2: Configuration-space trajectories for (a) the simple equilibrium-point solution with no payload, no learning, and no obstacle, (b) the "imitation" trial with no payload, (c) a representative solution with a 4.5 kg payload, and (d) an unexpected solution with an 9.25 kg payload. *S*, *V*, and *G* denote the start, via-point, and goal configurations, respectively. The SRS algorithm step size and search strategy parameters were set to $\alpha = 0.3$ and $\beta = 0$, respectively. For the motor program, the initial search size was $\sigma = 0.10$ with a minimum of $\sigma_{min} = 0.01$ and a decay factor of $\gamma = 0.95$. For the PD controllers, the corresponding parameter values were $\sigma = 0.05$, $\sigma_{min} = 0.01$, and $\gamma = 0.99$.

The SRS algorithm has several properties that make it a nice choice for trial-and-error learning. First, SRS is easy to implement for a wide variety of optimization problems. Like other direct search methods [Swann, 1972], the SRS algorithm needs no derivative information—an important feature when the cost function is non-differentiable or when the gradient is difficult to estimate (as with deterministic policies subject to noise). The algorithm also has some neurobiological support, albeit speculative [Anderson, 1998]. And compared to "pattern search" algorithms [Beveridge and Schechter, 1970], such as the simplex method, SRS makes rapid progress in high-dimensional spaces where some algorithms first require numerous exploratory moves to establish a search direction or to build a simplex, for instance.

4 Learning To Exploit Dynamics

With no payload, the robotic weightlifter can use a number of qualitatively different solutions to reach the goal configuration. For example, the simple equilibrium-point solution *with no obstacle* (Figure 2a) follows a direct path from the start to the goal, whereas the "imitation" trial (Figure 2b) takes a longer path through configuration space, first converging to the via point. As the payload increases, the space of feasible solutions shrinks, and so the via point represents an attempt to start the learning system at a favorable place. The "standard" solution after learning (Figure 2c) passes by the via point while coordinating the joints to exploit the robot's intrinsic dynamics. (The standard solution is robust to sensor noise as well as variability in the via point.) Figure 2 also shows an unexpected "reversal" solution, where the robot moves through an entirely different region of configuration space (details below).

As indicated in Section 3.2, each PD controller initially behaves as three independent, linear servomechanisms—one for each joint—but the learning algorithm transforms them into the robot analogue of a synergy [Bernstein, 1967; Greene, 1982] or "smart" nonlinear controller that accounts for intersegmental effects. (Although Eq. (1) describes a linear controller, saturation of the output represents a set of nonlinear constraints, possibly inactive, that the learning system can exploit.) To examine the coupling of the individual controllers, we quantify joint coordination as

$$C(\mathrm{PD}_1, \mathrm{PD}_2) = 2 - [D(\mathbf{W}_1) + D(\mathbf{W}_2)], \qquad (2)$$

where D is the following measure of diagonal dominance of a PD controller gain matrix:

$$D(\mathbf{W}) = \frac{\sum_j |w_{jj}|}{\sum_{j,k} |w_{jk}|} \qquad (3)$$

Figure 3 shows the change in C throughout learning—starting with no coupling (both $\mathbf{W}_1$ and $\mathbf{W}_2$ diagonal) for the initial trial and increasing to about 1.0 where half the "mass" in the gain matrices falls along the main diagonals. With no payload, the increase in coupling mirrors the drop in effort, and with all but the lightest payloads, we observe a statistically significant increase in coupling over the no-payload condition. Thus, active coordination by the control system appears to play an important role for the weightlifting task, although the functional significance of these results remains a question for future work.

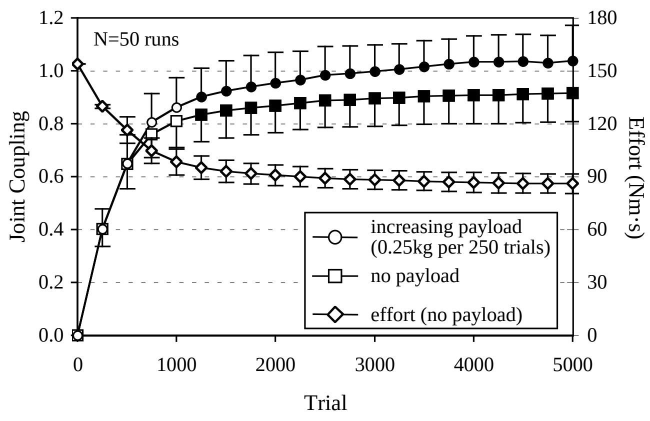

Figure 3: Effects of payload and learning on controller joint coordination (averaged over 50 runs). Solid markers denote statistically significant differences ($p<0.01$) at the corresponding trial number.

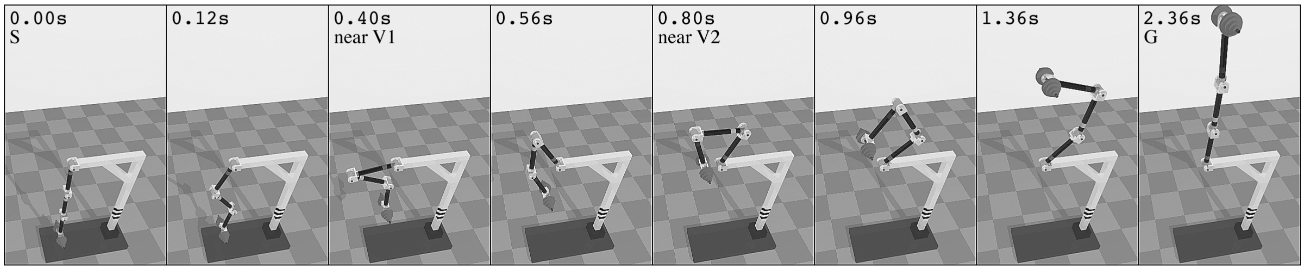

Figure 4: The reversal solution with a 9.25 kg payload. The manipulator was nearest the two via points, *V1* and *V2*, at *t*=0.40 s and *t*=0.80 s.

Early in learning, initial motion at the middle and distal joints induces counterclockwise movement at the proximal joint—despite the action of the PD controller which attempts to drive the upper arm clockwise, away from the support structure and directly toward the via point. In other words, passive mechanincal coupling conflicts with the initially uncoupled, active control. Through subsequent learning, however, the best solutions exploit intersegmental dynamics and initial swing (i.e., momentum) by reversing the sign of the "shoulder" torque for the first 100 to 200 ms.

On rare occasions—less than one percent of all runs—the learning system discovered the "reversal" solution shown first in Figure 2d and again, with more detail, in Figure 4. Throughout much of the movement, the positions of the middle and distal joints have opposite sign with respect to the standard solution. In effect, the reversal solution abandons the prior knowledge supplied by the via point. Interestingly, we observed the reversal solution only for those experiments with increased variability, involving sensor noise and a large initial search size, σ.

Once we know of an alternative solution, we can use the same approach to policy parameterization to offer new "coaching" to our robotic weightlifter. For instance, we can encourage initial counterclockwise swing by inserting an extra via point (and an extra PD controller) prior to the one shown in Figure 1b. Separately, we can also encourage the reversal solution by designing two new via points as depicted in Figure 4 (frames three and five). These alternatives resulted in statistically significant improvements ($p<0.001$) as summarized in Figure 5. Thus, by injecting simple, approximate knowledge, we turned the reversal solution from a rare occurrence into a new standard by which to gauge future performance.

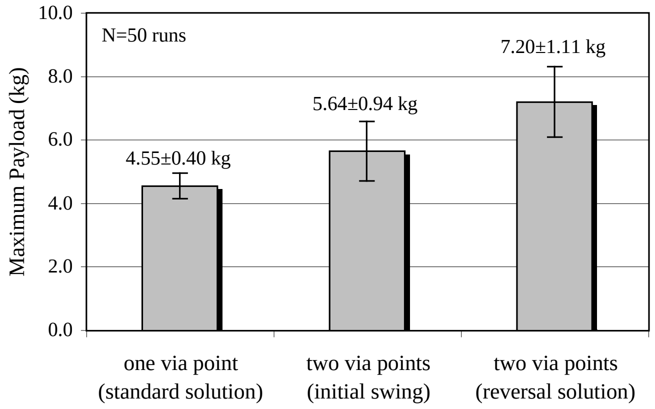

Figure 5: Effects of "coaching" on maximum payload lifted successfully after learning (averaged over 50 runs).

5 Conclusions

We attribute much of the success of the SRS algorithm to the prior structure supplied by both the parameterized policy and the via points. We suspect that more sophisticated forms of direct function optimization, e.g., [Swann, 1972], will yield improvements in terms of learning time, but that the qualitative results will remain the same. The more interesting possibility for future work is the use of gradient-based methods for continuing control tasks—whether a direct policy search algorithm, e.g., [Baxter and Bartlett, 2000], or a value-based approach, e.g., [Sutton *et al.*, 2000]. Such methods are particularly important for extending the basic approach outlined in this paper to non-episodic tasks with either occasional or continuing rewards.

For robotic manipulators, the classic solution to a movement problem involves, not reinforcement learning, but rather the construction of a kinematic trajectory that avoids obstacles and singularities. The underlying assumption is that the robot's intrinsic dynamics are something to compensate for, instead of something to exploit. Even for optimized trajectories, imprecise dynamic models often lead to overly conservative acceleration constraints and, therefore, to sub-optimal movements [Craig, 1988]. Rather than use adaptive control techniques to deal with inaccurate models, in this paper we gave up the stability guarantees associated with control engineering methods and developed a learning framework motivated by human motor control. As part of future work, we plan to test this framework on a real, non-planar robot with seven degrees of freedom. Like other reinforcement learning methods, our approach is geared toward optimization; thus, the resulting policies are inherently ones that exploit dynamics.

As mentioned in Section 3.1, one criticism of equilibrium-point styles of motor control is the need for virtual trajectories to explain complicated multi-joint movements. Van Ingen Schenau *et al.* [1995] also argue that equilibrium-point models are kinematic in nature and, therefore, ill-suited for tasks that exploit dynamics or require control of contact forces. Although we agree with these criticisms, equilibrium-point control nevertheless plays a key role in the present work. In particular, the movement primitives (i.e., the via points and the goal configuration) supply the hierarchical motor program with an initial kinematic solution that the learning algorithm then transforms into one that exploits, rather than cancels the dynamics.

Acknowledgments

We thank Theodore Perkins and Richard van Emmerik for comments and helpful discussions. This research was supported by the National Science Foundation under Grant No. IRI-9720345.

References

[Anderson, 1998] Russell W. Anderson. Biased random-walk learning: a neurobiological correlate to trial-and-error. In Omid Omidvar and Judith Dayhoff, editors, *Neural Networks and Pattern Recognition*, pages 221–244. Academic Press, San Diego, CA, 1998.

[Anderson, 2000] C. W. Anderson. Approximating a policy can be easier than approximating a value function. Technical Report CS-00-101, Colorado State University, 2000.

[Baird and Moore, 1999] L. Baird and A. Moore. Gradient descent for general reinforcement learning. In Michael S. Kearns, Sara A. Solla, and David A. Cohn, editors, *Advances In Neural Information Processing Systems 11*, pages 968–974. The MIT Press, Cambridge, MA, 1999.

[Baxter and Bartlett, 2000] Jonathan Baxter and Peter L. Bartlett. Reinforcement learning in POMDPs via direct gradient ascent. In *Proceedings of the Seventeenth International Conference on Machine Learning,*, 2000.

[Bernstein, 1967] N. A. Bernstein. *The Co-ordination and Regulation of Movements*. Pergamon Press, Oxford, 1967.

[Beveridge and Schechter, 1970] Gordon S. G. Beveridge and Robert S. Schechter. *Optimization: Theory and Practice*. McGraw-Hill Book Co., New York, 1970.

[Boutilier *et al.*, 2000] Craig Boutilier, Richard Dearden, and Moises Goldszmidt. Stochastic dynamic programming with factored representations. *Artificial Intelligence*, 121(1-2):49–107, 2000.

[Craig, 1988] John J. Craig. *Adaptive Control of Mechanical Manipulators*. Addison-Wesley Publishing Company, Reading, MA, 1988.

[Feldman, 1986] A. G. Feldman. Once more on the equilibrium-point hypothesis (λ model) for motor control. *Journal of Motor Behavior*, 18:17–54, 1986.

[Greene, 1972] P. H. Greene. Problems of organization of motor systems. In R. Rosen and F. M. Snell, editors, *Progress In Theoretical Biology*, volume 2, pages 303–338. Academic Press, New York, 1972.

[Greene, 1982] P. H. Greene. Why is it easy to control your arms? *Journal of Motor Behavior*, 14(4):260–286, 1982.

[Moore and Atkeson, 1995] A. W. Moore and C. G. Atkeson. The parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. *Machine Learning*, 21(3):199–233, 1995.

[Moriarty *et al.*, 1999] David E. Moriarty, Alan C. Schultz, and John J. Grefenstette. Evolutionary algorithms for reinforcement learning. *Journal of Artificial Intelligence Research*, 11:241–276, 1999.

[Morimoto and Doya, 1998] J. Morimoto and K. Doya. Reinforcement learning of dynamic sequence: learning to stand up. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 1721–1726, 1998.

[Schaal, 1999] Stefan Schaal. Is imitation learning the route to humanoid robots? *Trends in Cognitive Science*, 3:233–242, 1999.

[Schmidt, 1975] Richard A. Schmidt. A schema theory of discrete motor skill learning. *Psychological Review*, 82(4):225–260, 1975.

[Sutton *et al.*, 1999] Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112:181–211, 1999.

[Sutton *et al.*, 2000] Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In Sara A. Solla, Todd K. Leen, and Klaus-Robert Muller, editors, *Advances In Neural Information Processing Systems 12*, pages 1057–1063. The MIT Press, Cambridge, MA, 2000.

[Sutton, 1988] Richard S. Sutton. Learning to predict by the method of temporal differences. *Machine Learning*, 3:9–44, 1988.

[Swann, 1972] W. H. Swann. Direct search methods. In W. Murray, editor, *Numerical Methods For Unconstrained Optimization*, pages 13–28. Academic Press, New York, 1972.

[Tsitsiklis and Van Roy, 1997] J. N. Tsitsiklis and B. Van Roy. An analysis of temporal-difference learning. *IEEE Transactions on Automatic Control*, 42:674–690, 1997.

[van Ingen Schenau *et al.*, 1995] G. J. van Ingen Schenau, A. J. van Soest, F. J. M. Gabreels, and M. W. I. M. Horstink. The control of multi-joint movements relies on detailed internal representations. *Human Movement Science*, 14(4–5):511–538, 1995.

[Walker and Orin, 1982] Michael W. Walker and David E. Orin. Efficient dynamic computer simulation of robotic mechanisms. *Journal of Dynamic Systems, Measurement and Control*, 104:205–211, 1982.

[Wang *et al.*, 1999] C.-Y. E. Wang, W. K. Timoszyk, and J. E. Bobrow. Weightlifting motion planning for a puma 762 robot. In *Proceedings of the IEEE 1999 International Conference on Robotics and Automation*, volume 1, pages 480–485, 1999.

[Watkins and Dayan, 1992] C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3/4):279–292, 1992.